UCRL-PROP-145639DR

Attachment 2
DRAFT STATEMENT OF WORK
Version 9

ADVANCED SIMULATION AND COMPUTING

(ASCI)

PURPLE

B519700

LAWRENCE LIVERMORE NATIONAL LABORATORY

LIVERMORE, CALIFORNIA

February 15, 2002

Table of Contents

Definitions

Particular paragraphs of the Statement of Work have the following priority designations.

(a) **Mandatory requirements designated as (MR)**
Mandatory requirements are items that are essential to the University requirements and reflect the minimum qualifications an Offeror must meet in order to have their proposal evaluated further for selection.

(b) **Mandatory Option requirements designated as (MO)**
Mandatory Option requirements deal with features, components, performance characteristics, or upgrades whose availability as an option is deemed a Mandatory Requirement by the University. Hence, a proposal not meeting a Mandatory Option will be deemed technically nonresponsive. Because the University will variously elect to include or exclude such options in resulting orders, each should appear as a separately identifiable item in the "Alternate Proposals and Options" and "Price Proposal".

(b) **Target Requirements designated as (TR-1, TR-2 and TR-3).**
Each paragraph so labeled deals with features, components, performance characteristics or other properties that is considered a part of the ASCI system but will not be a determining factor of response compliance. Target requirements are prioritized by a dash number, TR-1 being the most important. Taken together the aggregate of the MR, MO and TR-1 requirements form a baseline system. TR-1 targets are as important to the program as mandatory requirements, but not meeting any particular TR-1 target requirement is insufficient to render a proposal as non-responsive. TR-2 targets are second priority after TR-1 requirements. TR-2 requirements are considered goals that boost a minimal baseline system, taken together as an aggregate of MR, MO, TR-1 and TR-2 requirements, into the moderately useful category. TR-3 targets are third priority after TR-2 requirements. TR-3 requirements are considered stretch goals that boost a moderately useful system, taken together as an aggregate of MR, MO, TR-1, TR-2 and TR-3 requirements, into the highly useful category. Thus, the ideal ASCI EDTV and Purple systems will meet or exceed all MR, MO, TR-1, TR-2 and TR-3 requirements. Target Requirement responses will be considered as part of the evaluation of Technical Proposal Excellence (see Attachment 3, Evaluation Criteria).

1.0 Introduction

1.1 The Stockpile Stewardship Program

The Stockpile Stewardship Program is managed by the National Nuclear Security Administration (NNSA). It is an essential part of the NNSA mission to carry out the Department of Energy (DOE) responsibility to achieve national security objectives established by the President for nuclear weapons. The NNSA is responsible for maintaining a safe, secure and reliable stockpile of nuclear weapons and associated materials, capabilities and technologies in a safe, environmentally sound, and cost effective manner.

The Stockpile Stewardship Program encompasses operations associated with manufacturing, maintaining, refurbishing, surveilling, and dismantling the nuclear weapons stockpile; activities associated with the research, design, development, simulation, modeling, and non-nuclear testing of nuclear weapons; and the planning, assessment and certification of safety and reliability. Many of these activities are performed at the three NNSA weapons laboratories (Lawrence Livermore National Laboratory, Sandia National Laboratories, and Los Alamos National Laboratory) and the Nevada Test Site.

On October 2, 1992, President Bush signed into law the FY1993 Energy and Water Authorization Bill that established a moratorium on U.S. nuclear testing. President Clinton extended the moratorium on July 3, 1993. These decisions ushered in a new era in the way the United States ensures confidence in the safety, performance, and reliability of its nuclear stockpile. The U.S. also decided to halt new nuclear weapon production. This decision meant that the U.S. stockpile of nuclear weapons would need to be maintained far beyond its design lifetime.

The Stockpile Stewardship Program was established to implement these pivotal policy decisions. The program must provide scientists and engineers with the technical capabilities to maintain a credible nuclear deterrent *without* the use of the two key tools used to do that job over the past fifty years: underground nuclear testing and modernization through development of new weapon systems.

Historically, U.S. policymakers were ensured confidence in the stockpile by the use of regular nuclear tests. They never had to rely on weapon systems that had exceeded their design-intent lifetimes because older weapons were regularly replaced with new designs. With the cessation of these two practices, the U.S. committed itself to maintaining its existing weapon systems indefinitely, well beyond their intended lifetimes. Implementing this policy with credibility requires new scientific tools. The responsibility to develop those tools resides with the Stockpile Stewardship Program.

To meet this challenge, a new set of aboveground, non-nuclear experimental capabilities was required, environmentally benign fabrication capabilities were needed, and archived data from decades of nuclear tests had to be made available to weapon scientists and engineers. An unprecedented level of computational capability was needed to serve as the integrating force to make effective use of the collective scientific understanding. This meant that a new and powerful role for modeling and simulation was required. The Accelerated Strategic Computing Initiative (ASCI), now the Advanced Simulation and Computing Program, was created to establish this capability.

1.2 Advanced Simulation and Computing (ASCI) Program Overview

The Advanced Simulation and Computing Program (ASCI) was established in 1995 as a critical element to help shift from test-based confidence to science- and simulation-based confidence. Specifically, ASCI is a focused and balanced program that is accelerating the development of simulation capabilities needed to analyze and predict the performance, safety, and reliability of nuclear weapons and certify their functionality—far exceeding what might have been achieved in the absence of a focused initiative.

To realize its vision, ASCI is creating simulation capabilities based on advanced weapon codes and high-performance computing that incorporate more complete scientific models based on experimental results, past tests, and theory. The result would be predictive simulations that enable assessment and certification of the safety, performance, and reliability of nuclear systems. These simulation capabilities will also help scientists understand weapons aging, predict when components will have to be replaced, and evaluate the implications of changes in materials and fabrication processes to the design life of the aging weapon systems. This science-based understanding is essential to ensure that changes brought about through aging or remanufacturing will not adversely affect the enduring stockpile.

To meet the needs and requirements of the Stockpile Stewardship Program, ASCI has specific program objectives in performance, safety, reliability, and sustainability:

- Performance: Create predictive simulations of nuclear weapon systems to analyze behavior and assess performance in an environment without nuclear testing.
- Safety: Predict with high certainty the behavior of full weapon systems in complex accident scenarios.
- Reliability: Achieve sufficient validated predictive simulations to extend the lifetime of the stockpile, predict failure mechanisms, and reduce routine maintenance.
- Sustainability: Use virtual prototyping and modeling to understand how new production processes and materials affect performance, safety, reliability, and aging. This understanding will help define the right configuration of production and testing facilities necessary for managing the stockpile throughout the next several decades.

ASCI must solve progressively more difficult problems as we move away from nuclear testing. To do this, applications must achieve higher resolution, higher fidelity, three-dimensional physics, and full-system modeling capabilities to reduce reliance on empirical judgments. This level of simulation requires high-performance computing far beyond our current level of performance. To accomplish this, ASCI is partnering with industry to accelerate development of more powerful computing hardware and is investing in creating the necessary software environment. A powerful problem-solving environment must be established to support application development and enable efficient and productive use of the new computing systems. By 2005, the ASCI program is responsible for the following deliverables:

- Development of high-performance, full-system, high-fidelity-physics predictive codes to support weapon assessments, renewal process analyses, accident analyses, and certification,
- Stimulation of the U.S. computer manufacturing industry to create the powerful high-end computing capability required by ASCI applications, and
- Creation of a computational infrastructure and operating environment that makes these capabilities accessible and usable.

The ASCI Program has, in fewer than five years, produced results that may well make it the most successful high-performance computing program in U.S. history. Five of the top eight systems on the Top 500 of the world's fastest computers are the ASCI White and ASCI Blue-Pacific at Lawrence Livermore National Laboratory, Blue Mountain and Q at Los Alamos National Laboratory, and ASCI Red at Sandia National Laboratories. These systems in turn have been instrumental in first-time, three-dimensional simulations involving components of a nuclear weapon during an explosion. Such accomplishments are based on the successes of other elements of ASCI research, such as scalable algorithms, programming techniques for thousands of processors, and unparalleled visualization capabilities. These accomplishments offer confidence that the challenging goals and objectives facing the program can be achieved.

ASCI is organized around major program elements. Each element has top-level strategies to contribute to the overall program goals. The following list gives the integrated program elements organized into five categories.

Defense Applications and Modeling

- Advanced Applications Development
  - Accelerated development of higher performance software to implement three-dimensional, high-fidelity-physics simulation and prototyping
- Verification and Validation (V&V)
  - Achieving high confidence in the computational accuracy of ASCI codes and their underlying models
- Materials and Physics Modeling
  - Promoting capabilities to predict the physical properties of matter under conditions found in nuclear explosions
  - Development of underpinning physics and prediction of material properties under Stockpile-to-Target Sequence (STS) environments from fabrication through nuclear explosion

Simulation and Computer Science

- Problem Solving Environment (PSE)
  - A computational infrastructure enabling execution of ASCI applications and access from the desktops of scientists
- Visual Interactive Environment for Weapons Simulation (VIEWS)
  - Developing "see and understand" technologies for ASCI simulations, enabling scientists to view the three-dimensional results of their stewardship calculations
- Distance and Distributed Computing and Communication ($DisCom^2$)
  - Supporting high-end computing to remote sites and an integrated computing environment distributed across the nuclear weapons complex
- PathForward
  - Accelerating commercial development of technologies needed for 60 teraFLOP/s platforms and beyond

Integrated Computing Systems

- Physical Infrastructure and Platforms
  - Developing powerful ASCI platforms in partnership with industry
- Ongoing Computing
  - Operating ASCI supercomputers

University Partnerships

- Academic Strategic Alliances Program (ASAP)
  - Engaging academia to accelerate simulation science
- Institutes and Fellowships
  - Involving the U.S. academic community in ASCI science and computing

Program Integration

- One Program – Three Laboratories
  - Planning and implementation of all ASCI efforts conducted in concert with participation from ASCI Headquarters and the three national security laboratories

As an integral and vital element of the Stockpile Stewardship Program, ASCI provides the integrating simulation and modeling capabilities and technologies needed to combine new and old experimental data, past nuclear test data, and past design and engineering experience into a powerful tool for future design assessment and certification of nuclear weapons and their components. ASCI capabilities are needed to model prior manufacturing processes for weapon components and define new, cost-effective, safe, and environmentally compliant manufacturing processes that will provide for consistent nuclear weapon performance, safety, and reliability in the future.

The simulation and modeling tools have already made impacts on the assessment of stockpile issues. Weapon designers, scientists, and engineers are applying ASCI simulation and modeling capabilities and technologies to assess aging changes occurring in stockpile nuclear weapons and to assess and certify planned refurbishments of weapon system components.

From the beginning of the program, ASCI has been focused on achieving a 100 teraFLOP/s machine to provide the minimum capability required to begin to address the significant integrated application requirements of the program. ASCI has had a well-defined and delineated roadmap, and has met all of its milestones for delivery of machines up to and including the 12.3 teraFLOP/s White machine at Lawrence Livermore. The 30 teraFLOP/s Q machine will be sited at Los Alamos. The present machine procurement is intended to continue along this roadmap, subject to program and budgetary constraints.

1.3 Current ASCI Platforms

Over the last five years, the ASCI Program has acquired several computational platforms: the Red machine at Sandia National Laboratories; the Blue Pacific and White systems at Lawrence Livermore National Laboratory; and the Blue Mountain and Q platforms at Los Alamos National Laboratory. The following information summarizes the existing generation of ASCI Platforms, as well as other platforms used by DOE. A more complete description of the systems can be found at URL http://www.llnl.gov/asci/platforms/

1.3.1 ASCI Red

The ASCI Red system, provided by the Intel Corporation, is a distributed memory, multiple instruction/multiple data (MIMD), message-passing machine. All aspects of the system architecture are scalable, including communication bandwidth, main memory, internal disk storage, capacity, and I/O. It contains approximately 9,000 Intel 333 MHz Pentium II Xeon processors, 1.2 TiB of system RAM, 12.5 TB of global disk and has a peak performance of 3.15 TF. A complete, but out of date, description of the system can be found at URL: <http://www.sandia.gov/ASCI/Red/>

1.3.2 ASCI Blue-Pacific

The ASCI Blue-Pacific system, provided by IBM, is based upon the RS/6000 SP computer with Silver nodes. The final configuration of the Blue Pacific system consists of 5,856 IBM PowerPC 604e CPUs. Its peak performance rating is 3.9 TF, 2.5 TiB of memory and 75 TB of global disk. A complete description of the system can be found at URL: http://www.llnl.gov/asci/platforms/bluepac/

1.3.3 ASCI White

The ASCI White system is a follow-on to the Blue-Pacific system. It is composed of 512 IBM RS/6000 NightHawk-2 16-way SMP nodes using 375 MHz IBM 64-bit POWER3-II CPUs and has peak performance rating of 12.3 TF, 8 TiB of memory and 122 TB of global disk. A more complete description of the system can be found at URL: http://www.llnl.gov/asci/platforms/white/

1.3.4 ASCI Blue Mountain and Nirvana

The ASCI Blue Mountain system, provided by Silicon Graphics, Inc. (SGI), is composed of a cluster of Origin2000 SMPs. Blue Mountain consists of 6,144 MIPS R10000 CPUs, 1.5 TiB of memory, and 76 TB of disk. Its peak performance rating is 3.072 TF. ASCI Blue Mountain operates in the classified portion of the Los Alamos computing environment. The Nirvana system operates in the unclassified portion of the Los Alamos computing environment. It is identical to the Blue Mountain system in its technology and software, but is smaller in scale. The Nirvana system is composed of 2,048 MIPS R10000 CPUs, 512 GiB of RAM, and 6.912 TB of disk. Its peak performance rating is approximately 1 TF. A more complete description of the system can be found at URL: http://www.lanl.gov/asci/bluemtn/

1.3.5 ASCI Q

The ASCI Q platform is currently under development and deployment to LANL by Compaq Computer Corporation. Q is targeted for 30 teraFLOP/s performance level in the 2H2002. More information is available at URL: http://www.compaq.com/hpc/tsn/iss017/hptc_iss017_fa.html

1.4 ASCI Applications Overview

ASCI applications codes will generally perform complex time-dependent three-dimensional simulations of multiple physical processes, where often the processes are tightly coupled and will require physics models linking microscale phenomena to macroscopic response. These codes will address nuclear weapons issues relating to safety, performance, reliability, and sustainability, which are of interest to the NNSA Weapons Laboratories in support of Stockpile Stewardship. The ASCI codes will include multi-material shock hydrodynamics, radiation and particle transport, atomic physics, material properties, and structural response.

Demanding stockpile stewardship issues are driving intense development of ASCI applications codes. Advanced numerical algorithms require innovative software techniques to achieve the necessary delivered performance on advanced computing platforms. Current application characteristics and expected trends are described below, although ASCI applications codes and numerical algorithms will continue to evolve rapidly.

Applications development is focused on a series of high level Milestones ("Level-1 Milestones," previously referred to as Mileposts), in addition to the larger number of project milestones and activities

that support directed stockpile work. The Milestones related to nuclear, safety, and non-nuclear applications are summarized in the following Table. The non-nuclear application Milestones involve simulations of the Stockpile-to-Target-Sequence (STS). Note particularly the application Milestones to be accomplished during the anticipated time frame for the operation of Option Purple. Additional Level-1 Milestones, not shown here, have been defined for the other elements of the ASCI Program.

ASCI Application Mileposts	**Fiscal Year/Quarter**
3D Primary-Burn Prototype Simulation	Completed FY00/Q1
3D Prototype Hostile-Environment Simulation (Non-Nuclear Application)	Completed FY00/Q2
3D Secondary-Burn Prototype Simulation	Completed FY01/Q1
STS Normal Environment Prototype Simulation (Non-Nuclear Application)	FY01/Q4
3D Prototype Full-System Coupled Simulation	FY02/Q1
STS Abnormal Environment Prototype Simulation (Non-Nuclear Application)	FY02/Q4
3D Safety Simulation of a Complex Abnormal Explosive-Initiation Scenario (Safety Application)	FY02/Q4
3D High-Fidelity-Physics Primary-Burn Simulation, Initial Capability	FY03/Q1
Coupled STS Hostile Environment Simulation (Non-Nuclear Application)	FY03/Q4
3D High-Fidelity-Physics Secondary-Burn Simulation, Initial Capability	FY04/Q1
3D High-Fidelity-Physics Full-System Simulation, Initial Capability	FY04/Q4
Full System STS Simulation (Non-Nuclear Application)	FY05/Q4

The following are some of the major ASCI code characteristics essential to an understanding of the vision of an ideal computing environment.

The codes model multiple types of physics, generally in a single (usually monolithic) application, in a time-evolving manner with direct coupling between all simulated processes. They do so using a variety of computational methods, often through a separation or "split" of the various physics computations and coupling terms. This process involves doing first one type of physics, then the next, then another, and then repeating this sequence for every time step. Some algorithms are categorized as explicit in time while others are fully implicit or semi-implicit and typically involve iterative solvers of some form. Some special wavefront "sweeps" are employed for specific direct-solve algorithms. Numerous research efforts are actively exploring novel and alternative methods and algorithms for possible application to problems of interest.

The calculations treat millions of spatial zones or cells, with an expected requirement for many applications to get to the point of using about a billion zones. The equations are typically solved by spatial discretization. Discretization over energy and/or angle, in addition, can increase the data space size by 10 to 1000 times. In the final analysis, thousands of variables will be associated with each zone. Monte Carlo algorithms will treat millions to billions of particles distributed throughout the problem domain. The parallelization strategy for many codes is based upon decomposition into spatial domains. Some codes will use decomposition over angular or energy domains, as well, for some applications.

Currently, almost all codes use the standard message passing interface (MPI) for parallel communication, even between processes running on the same SMP. In addition, some applications utilize OpenMP for SMP parallelism. The efficiency of OpenMP SMP parallelism depends highly on the underlying compiler implementation (i.e., the algorithms are highly sensitive to OpenMP overheads). Also, it is

possible in the future that different physics models within the same application might use different communication models. For example, an MPI-only main program may call a module that uses the same number of MPI processes, but also uses threads (either explicitly or through OpenMP). In the ideal system, these models should interoperate as seamlessly as possible. Mixing such models mandates thread-safe MPI libraries. Alternative strategies may involve calling MPI from multiple threads with the expectation of increased parallelism in the communications; such use implies multi-threaded MPI implementations as well.

Current codes are based on a single program multiple data (SPMD) approach to parallel computing. However, director/worker constructs are often used. Typically, data are decomposed and distributed across the system and the same execution image is started on all MPI processes and/or threads. Exchanges of remote data occur for the most part at regular points in the execution, and all processes/threads participate (or just pretend to) in each such exchange. Data are actually exchanged with individual MPI send-receive requests, but the exchange as a whole can be thought of as a "some-to-some" operation with the actual data transfer needs determined from the decomposition. Weak synchronization naturally occurs in this case because of these exchanges, while stronger synchronization occurs because of global operations, such as reductions and broadcasts (e.g., MPI_allreduce), which are critical parts of iterative methods. It is quite possible that future applications will use functional parallelism, but mostly in conjunction with the SPMD model. Parallel input-output (I/O) and visualization are areas that may use such an approach with functional parallelism at a high level to separate them from the physics simulation, yet maintain the SPMD parallelism within each subset. There is some interest in having visualization tools dynamically attach to running codes and then detach for interactive interrogation of simulation progress. Such mixed approaches are also under consideration for some physics models.

Many applications use unstructured spatial meshes. Even codes with regular structured meshes may have unstructured data if they use cell-by-cell continuous adaptive mesh refinement (AMR). In an unstructured mesh, the neighbor of zone (i) is not zone (i+1), and one must use indirection or data pointers to define connectivity. Indirection has been implemented in several codes through libraries of gather-scatter functions that handle both on-processor as well as remote communication to access that neighbor information. The communication support is currently built on top of MPI and/or shared memory to get that neighbor information. These scatter-gather libraries are two-phased for good efficiency. In phase one, the gather-scatter pattern is presented and all local memory and remote memory and communications structures are initialized. Then in phase two, the actual requests for data are made, usually many, many times. Thus, the patterns are extensively reused over and over again. Also, several patterns will coexist simultaneously during a timestep for various data. Techniques like AMR and reconnecting meshes can lead to pattern changes at fixed points in time, possibly every cycle or maybe only after several cycles.

Memory for arrays and/or data structures is typically allocated dynamically, avoiding the need to recompile with changed parameters for each simulation size. This allocation requires compilers, debuggers, and other tools that recognize and support such features as dynamic arrays and data structures, as well as memory allocation intrinsics and pointers in the various languages.

Many of the physics modules will have low compute–communications ratios. It is not always possible to hide latency through non-blocking asynchronous communication, as the data are usually needed to proceed with the calculation. Thus, a low-latency communications system is crucial.

Many of the physics models are memory intensive, and will perform only about 1 FLOP per load from memory. Thus, performance of the memory sub-system is crucial, as are compilers that optimize in terms of cache blocking, loop unrolling-rolling, loop nest analysis, etc. Many codes have loops over all points in an entire spatial decomposition domain. This coding style is preferred by many for ease of implementation and readability of the physics and algorithms. Although recognized as problematic, effective automatic optimization is preferred, where possible.

The multiple physics models embedded in a large application may have dramatically varying communication characteristics, i.e., one model may be bandwidth-sensitive, while another may be latency-sensitive. Even the communications characteristics of a single physics model may vary greatly during the course of a calculation as the spatial mesh evolves or different physical regimes are reached and the modeling requirements change. In the ideal system, the communications system should handle this disparity without requiring user tuning or intervention.

Although static domain decomposition is used for load balancing as much as possible, there are also definite needs for dynamic load balancing, in which the work is moved from one processor to another. One obvious example is for codes using AMR methods, where additional cells may be added or removed during the execution wherever necessary in the mesh. It is also expected that different physical processes will be regionally constrained and, as such, will lead to load imbalances that can change with time as different processes become "active" or more difficult to model. Any such dynamic load balancing is expected to be accomplished through associated data migration explicitly done by the application itself. This re-balancing might occur inside a time step, every few timesteps, or infrequently, depending on the nature of the problem being run. In the future, code execution may also spawn and/or delete processes to account for the increase and/or decrease in the total amount of work the code is doing at that time.

1.5 ASCI Software Development Environment

The following are some of the major characteristics of the software development environment in an ideal scenario.

A high degree of code portability and longevity is a major objective. ASCI codes must execute at all three ASCI sites located at Lawrence Livermore National Laboratory, Sandia National Laboratories and Los Alamos National Laboratory. Development, testing and validation of 3D, full-physics, full system applications requires four to six years. The productive lifespan of these codes is at least ten years. Thus these applications must span not only today's applications but any possible future system. Codes will be developed in standards-conforming languages so non-standard compiler features are of little interest unless they can be made transparent. The use of Cray Pointers in Fortran is an exception to our reliance on standard features. We also will not take advantage of any idiosyncratic features of optimization, unless they can be hidden from the codes (e.g., in a standard library). Non-standard "hand tuning" of codes for specific platforms is antithetical to this concept.

A high-performance, low-latency MPI environment that is robust and scalable is crucial to us. Today applications are utilizing all the features of MPI 1.1 functionality. Many features of MPI-2 functionality are also in current use. Hence, a full, robust and efficient implementation of MPI-2 is of tremendous interest. A POSIX compliant-thread environment is also crucial and a Fortran95-threads interface is also important. All libraries need to be thread-safe. MPI should be multi-threaded as well as thread-safe. We should not have to tune the MPI-runtime environment for different codes and different problem sizes. In

our estimation, bandwidth of 0.2 bytes/second/peak OP/second/SMP and an end-to-end MPI ping-pong latency of less than 10 microseconds or better would provide the desired performance. Since we are talking about systems with tens of thousands of processors, it is vitally important that the MPI implementation scale to the full size of the system. This scaling is both in terms of efficiency (particularly of the MPI_ALLREDUCE functionality) as well as the efficient use of buffer memory. ASCI applications are carefully programmed so that MPI RECIEVE operations are posted before the corresponding SEND operation. This allows for minimal (and hence scalable) MPI buffer space allocations.

ASCI applications require the ability for each MPI task to access all physical memory on a node. The large memory sizes of MPI tasks requires that all of our applications need to be completely 64b by default.

We would expect the compilers to do the vast majority of code optimization through simple easy-to-use compiler switches (e.g. -On). Also, we would expect the compilers to have options to range check arrays under debug mode, as well as a way to trap underflow, overflow, divide by zero, etc. Parallelization through the OpenMP specifications is of particular interest and is expected for Fortran95, C, and C++. OpenMP parallelization must function correctly in programs that also use MPI. OpenMP Version 2.0 support for Fortran95, Version 1.0 for C/C++ is highly favored, while automatic parallelization is of some interest, if it is efficient and does not drive compile times to unreasonable lengths. Any information the compiler can provide about the optimizations it performed is useful. Compiler parallelism has to work in conjunction with MPI. All compilers must be fully ANSI-compliant.

The availability of standard, platform-independent tools is necessary for a portable and powerful development environment. Examples of these tools are GNU software (especially GNU make, but others as well), TotalView debugger (the current debugger on all ASCI platforms), dependency builders (Fortran USE & INCLUDE as well as #include), preprocessors (CPP, M4), source analyzers (lint, flint, etc), hardware counter libraries and communications profilers (VAMPIR, etc). Tools that work with a source code should fully support the most current language standards. A standard API to give debuggers and performance analyzers access to the state of a running code would allow us to develop our own tools or to use a variety of tools developed by others. The Distributed Process Class Library (DPCL) is an emerging public domain API that meets this need. These performance and debugging tools must not require privileged access modes, such as root user nor compromise the security of the runtime environment.

We must have parallel debuggers that allow us to debug parallel applications within an SMP or node and that permit parallel application debugging applications utilizing multiple nodes or SMPs. This includes MPI-only as well as mixed MPI + threads and/or OpenMP codes. In the best of all possible worlds, the debugger would allow effective debugging of jobs using every CPU on the system. Practical use of large fractions of the machine by an application under the control of the debugger requires that the debugger be highly integrated into the system initiated parallel checkpoint/restart and GANG scheduling mechanisms. Some specific features of interest include the following:

- breakpoints,
- fast conditional breakpoints,
- fast conditional watchpoints on memory locations,
- a save-restore state for backing up via checkpoint/restart mechanism,
- complex selections for data display (possibly even programming support with loops, conditionals, local variables, etc),

- support for array statistics (min, max, etc),
- attaching/detaching to running jobs,
- an initialization file that knows where the sources are and what options we want etc., and
- a command-line interface in addition to a GUI (e.g. for working over slow phone lines from home).

The capability to visually examine slices and subsets of multidimensional arrays is a feature that has proven useful. The debugger should allow complex selections for data display to be expressible with Fortran95 and C language constructs and features. It should support applications written in a mixture of the baseline languages (Fortran95, C and C++), support Cray-style pointers in Fortran77, and be able to dive on templated functions and handle complex template evaluation capabilities in C++. It should be able to debug compiler-optimized code since problems sometimes go away at debug levels, although less symbolic and contextual information will be available to the debugger at higher levels of optimization. Our build environment involves accessing source code from NFS and/or NFSv4 mounted file systems with likely compiling and linking of the executable in alternate directories. This process may have implications, depending on how the compiler tells the debugger to find the source code. The debugger currently used in the Tri-Laboratory ASCI PSE CDE is the TotalView debugger from Etnus (see URL: http://www.etnus.com/Products/TotalView/index.html) .

Because most ASCI codes are memory-access intensive, optimizing the spatial and temporal locality of memory accesses is crucial for all levels of the memory hierarchy. To tune memory distribution in a NUMA machine, it is necessary to be able to specify where memory is allocated. To optimally use memory and to reuse data in cache, it is also necessary to cause threads to execute on CPUs that quickly access particular NUMA regions and particular caches. Expressing such affinities should be an unprivileged operation. Threads generated by a parallelizing compiler (OpenMP or otherwise) should be aware of memory-thread affinity issues as well.

Other ramifications of the large memory footprint of ASCI codes is that they require full 64b support in **all** supplied hardware and software. In addition, because these memory-access intensive codes have random memory access patterns (due to indirect addressing or complex C++ structure and method dereferencing brought about from implementing discretization of spatial variables on block structured or unstructured grids) and hence access thousands to millions of standard UNIX™ 4KB VM pages every timestep, “large page support” in the operating system for efficient utilization of the microprocessor virtual to real memory translation functionality and caches is required for efficient use of the hardware. This is due to the fact that hardware TLBs have a limited number of entries (although caching additional entries in L1 cache helps but does not solve the problem) and having, say, 2GiB page size would significantly reduce the number of TLB entries required for large memory-access ASCI code VM to real memory translations. Since TLB misses (that are not cached in L1) are very expensive, this feature can significantly enhance ASCI application performance.

Many of our codes could benefit from a high-performance, standards-conforming, parallel I/O library, such as MPI-I/O. Many ASCI applications development teams now consider the ability to do MPI-2 dynamic tasking an essential item for future ASCI code development efforts. In addition, low latency GET/PUT operations for transmission of single cache lines is viewed as essential for domain overloading on a single SMP or node. However, many implementations of the MPI-2 MPI_GET/MPI_PUT mechanisms do not have lower latency than MPI_SEND/MPI_REC, but do allow for multiple outstanding MPI_GET/MPI_PUT operations to be active at a time. This approach although appealing to MPI-2 library developers, put the onus of latency hiding on the applications developer, who would rather

think about physics issues. Future ASCI applications require a very low latency (as close to the SMP memory copy hardware latency as possible) for GET/PUT operations.

It is advantageous to have support for translating big-endian, little-endian, and Cray Research PVP data representations to the system's native data forms. Especially useful would be automatic I/O filters on a file-by-file basis that would do this at read-write time.

A staff of Q-cleared, on-site applications/software analysts should be available to directly support code development activities of the major ASCI code projects. Ideally, these analysts should have a background in physics, numerical mathematics, or other hard computational oriented science and possess parallel computing and scientific applications development experience. The analysts should be closely associated with the software development organization and, in a sense, be an extension of the development team. Our experience has been that such analysts are critical to our ability to make progress on applications codes on complex ASCI scale systems. The importance of their role cannot be overemphasized.

Effectively tuning an application's performance requires detailed information on its timing and computation activities. On an SMP or node, a timer should be provided that is consistent between threads or tasks running on different CPUs in that same SMP or node. The timer should be high-resolution (10 microseconds or better) and low overhead to call. In addition, other hardware performance monitoring information such as the number of cache misses, TLB misses, floating-point operations, etc. can be very helpful. All modern microprocessors contain counters that gather this kind of information. The data in these counters should be made available separately for each thread or process through tools or programming libraries accessible to the user. For portability, our tools are targeting the PAPI library for hardware counters (http://icl.cs.utk.edu/projects/papi/). To limit instrumentation overhead, the vendor should provide a version of their tools that support multiplexing of hardware counters, and sampling of instructions in the pipeline. Note that this facility requires that the operating system context switch these counters at process or heavy weight (OS scheduled) thread level and that the POSIX or OpenMP runtime libraries context switch the counters on light weight (library scheduled) thread level. Furthermore, these counters must be available to users that do not have privileged access, such as the root user. Per-thread OS statistics must be available to all users via a command line utility as well as a system call. One example of such a feature is the `kstat` facility: a general-purpose mechanism for providing kernel statistics to users. Both hardware and counter statistics must provide virtualized information, so that users can make the correct attribution of performance data to application behaviors.

We need to have early access to new versions of system and development software, as well as other supplied software. Software releases of the various products should be synchronized with operating system releases to ensure compatibility and interoperability.

The Early Deployment of Technology Vehicle (EDTV), discussed elsewhere in this document, is important to us. It would be made available and supported for ASCI code development and testing, including system hardware and software upgrades as appropriate.

1.6 ASCI Applications Execution Environment

The following are some major characteristics of the ASCI ultra-scale applications execution environment.

It is crucial to be able to run a single parallel job on the full system using all resources available for a week or more at a time. This is called a "full-system run." Any form of interruption should be minimized. The capability for the system and application to "fail gracefully" and then recover quickly and easily is an extremely important issue for such calculations. We also expect to be running a large number of jobs on thousands of processors each for hundreds of hours. These would require significant system resources, but not the entire system. The capability of the system to "fail gracefully," so that a failure in one section of the system would only affect jobs running on that specific section, is important. From the applications perspective, the probability of failure should be proportional to the fraction of the system utilized. A failed section should be repairable without bringing down the entire system.

A single simulation may run over a period of months as separate restarted jobs in increments of days running on varying numbers of processors with different physics models activated. Output files produced by a code on one set of processors need to be efficiently accessible by another set of processors, or possibly even by a different number of processors, to restart the simulation. Thus an efficient cluster wide file system is essential. Ideally, file input and output between runs should be insensitive to the number of processors before and after a restart. It should be possible to restart a job across a larger or smaller number of processors than originally used, with only a slight difference in performance visible.

ASCI applications write many restart and visualization dumps during the course of a run. A single restart dump would be about the same size as the job's memory image, while visualization dumps would be perhaps from 1 to 10 % of that size. Restart dumps would typically be scheduled based on wall clock periods, while visualization dumps are scheduled entirely on the basis of internal physics simulation time. We usually create visualization dumps more frequently than restart dumps. System reliability will have a direct effect on the frequency of restart dumps; the less reliable the system is, the more frequently restart dumps will be made and the more sensitive we will be to I /O performance. We have observed on previous generation ASCI platforms that restart dumps comprise over 75% of the data written to disk. Most of this I/O is wasted in the sense that restart dumps are overwritten as the simulation progresses. However, this I/O must be done so that the simulation is not lost to a platform failure. This leads us to the notion that cluster wide file system (CWFS) I/O can be segregated into two portions: productive I/O and defensive I/O. Productive I/O is the writing of data that the user needs to do science (visualization dumps, traces of key physics variables over time, etc.). Defensive I/O is done to manage a large simulation run over a period of time much larger than the platform MTBF. Thus, one would like to minimize the amount of resources devoted to defensive I/O and computation lost due to platform failure.

In addition, while a full-system run is active, other applications development efforts will be engaged in preparation for their full-scale runs. This requires short access with fast response time to portions of the capability resource. Hence a system initiated parallel checkpoint/restart and GANG scheduling functionality is an essential ingredient to efficient operation of ASCI ultra-scale platforms.

System (hardware and software) failure modes should be clear and unambiguous. Supplied software should detect hardware and system software failures, report the error in a clear and concise manner to user as well as system administrator as appropriate, and recover with minimal to no impact to applications whenever possible.

Operationally, applications teams push the large restart and visualization dumps (already described) off to HPSS tertiary storage within the wall clock time between making these dumps. The disk space

mentioned elsewhere in this document is insufficient to handle ASCI applications long-term storage needs. HPSS is the archive storage system of ASCI and compatibility with it is needed. Thus, a highly usable mechanism is required for the parallel high speed transport of 10's to 100's of TB of data from the CWFS to HPSS.

We need a resource manager-job scheduler that deals with all aspects of the system's resources, not with just the processors and the time allocations. Factors that should be considered include processors, processes, memory, interconnects, disks, visualization engines, etc. It would be essential for this resource manager-scheduler to handle both batch and interactive execution of both serial and parallel programs (MPI and threaded) from a single processor to the full cluster. The manager-scheduler would provide a way to implement policies on selecting and executing various problems (problem size, problem run time, timeslots, preemption, users' allocated share of machine, etc). Also, a method would be provided for users to connect to executing batch jobs to query or change problem status or parameters. The tool(s) would schedule jobs to provide for process-to-processor affinity. We are currently using Platform Computing's Load Sharing Facility (LSF) on the ASCI Blue Mountain system and LLNL's DPCS on the ASCI Blue Pacific and White systems.

Our codes and users would benefit from a robust, globally visible, high-performance, parallel file system. It is essential that all file systems and software support 64b address space. A 32b address space is clearly insufficient.

A Q-cleared, on-site staff of software analysts would be available to support both the system software and users as consultants to the local staff.

The system would be composed primarily of commercial, commodity, product-line hardware. By this we mean that the essential building blocks of the system consist of the company's standard commercial hardware, as opposed to a completely custom-designed and fabricated one-of-a-kind system. We realize, however, that some components, by necessity, will be unique, as will the size and configuration. The intent, however, is to leverage the commodity marketplace as much as possible.

1.7 ASCI EDTV/Purple Operational Environment

The system would be designed to minimize floor space, power, and cooling requirements.

We plan to operate the systems 24 hours per day, 7 days per week, including holidays. The prime shift will be from 8 AM to 5 PM, Pacific Time Zone. LLNL local and remote (e.g., LANL and SNL) users would access the system via the 1 and 10 Gigabit Ethernet local-area network (LAN). For remote users, the Purple 1 and 10 Gigabit Ethernet infrastructure will be switched to the DisCom2 wide-area network (WAN) which will be OC-48/ATM/ POS connections.

The prime shift period will be devoted primarily to interactive applications development, interactive visualization, relatively short large CPU count (e.g., over half the system CPUs), high priority production runs and extremely long running, smaller CPU count (e.g, 1,024-2,048), lower priority production runs. Night shifts, as well as the weekend and holiday periods, would be devoted to extremely long-running jobs. Checkpointing and restarting jobs would take place as necessary to schedule this heterogeneous mix of jobs under dynamic load and job priorities on Purple. Because the workload is so varied and the demands for CPU time oversubscribe the machine by several factors, GANG scheduling utilizing the checkpoint/restart facility to **dynamically time-share** as well as space-share Purple is an essential

production requirement. In addition to system initiated checkpoint/restart, ASCI applications have the ability to do application based restart dumps. These interim dumps, as well as visualization output, would be stored on HPSS-based archival systems or sent to the VIEWS visualization corridors via the system-area network (SAN) and external "Jumbo Frame" 1 and 10 Gigabit Ethernet interfaces. Depending upon system protocol support, IP version 4, IP version 6, and lightweight memory-to-memory protocol (e.g., Scheduled Transfer) traffic will be carried in this environment.

Hardware maintenance services would be required around the clock, with two hour response time during the hours of 8:00 a.m. through 5:00 p.m., Monday through Friday (excluding Laboratory holidays), and less than four hours response time otherwise. The following are those holidays currently observed by the University at LLNL:

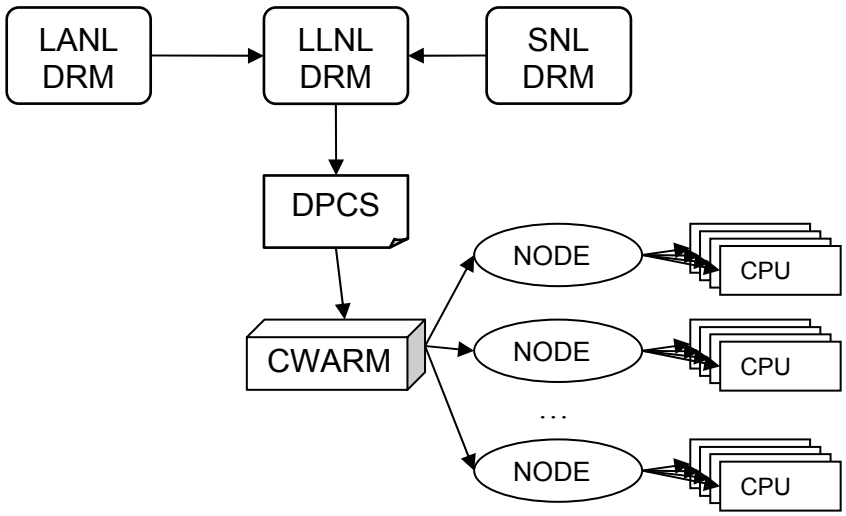
- New Year's Day
- Martin Luther King, Jr., Day (third Monday in January)
- President's Day (third Monday in February)
- Memorial Day (last Monday in May)
- Fourth of July
- Labor Day
- Thanksgiving Day
- Friday following Thanksgiving Day
- December 24 (or announced equivalent)
- Christmas Day
- December 31 (or announced equivalent)
- One administrative holiday to be selected by the President of the University

A single point of system administration would allow the configuration of the entire system from a common server. The single server would control all aspects of system administration in aggregate. Examples of system administration functions include modifying configuration files, editing mail lists, software, upgrades and patch (bug fix) installs, kernel parameter changes, file system-disk manipulation, reboots, user account activities (adding, removing, modifying), performance analysis, hardware changes, and network changes. A hardware and software configuration management system that profiles the system hardware and software configuration as a function of time and keeps track of who makes changes is essential.

Due to the large size of Purple, it is anticipated that the Offeror's System Test facility may not always be able to test software releases and bug fixes at scale. Although it is expected that a rigorous and intelligent testing mythology will always be employed by the Offeror prior to delivery of system releases or bug fixes, the final step in scaling and performance testing might, at times, have to be accomplished on Purple. Although this use of the system by the Offeror should be kept to an absolute minimum, there will be times when new releases and or patches will need to be installed on an interim basis on Purple. To this end we require a dual boot capability on the machine so that the known good, production quality software environment is not disrupted by the new releases and or bug fixes. This dual boot capability should be sufficient to bring the system to the new software level quickly and return the system to the previous state quickly after the debug shot. This of course engenders a requirement for fast and reliable full system reboot as it does not make sense to most sentient beings to have a four hour debug shot and an eight to sixteen hour period for the minimum of two required system reboots (one to boot the test system and one to boot the production system, assuming each reboot is successful on the first attempt).

The ability to dynamically monitor system functioning in real time and allow system administrators to quickly diagnose hardware, software (e.g., job scheduler) and workload problems and take corrective action is also essential. Due to the anticipated large size of Purple, these monitoring tools must be fast, scalable and display data in a hierarchal schema. The overhead of system monitoring and control will necessarily need to be low in order to not destroy large job scalability (performance).

At the highest level, the workload will be managed by the ASCI tri-laboratory inter-site Distributed Resource Manager (DRM). The primary purpose of the DRM is to provide a common tri-laboratory user interface to distribute resources to users, or groups of users, within a parallel distributed computing environment. At LLNL the DRM will interface into the Distributed Production Control System (DPCS). The DPCS system will control the use of the resources for both interactive and batch usage from a single CPU to all CPUs in compute SMPs in the system utilizing the Offeror supplied Cluster Wide Accounting and Resource Management (CWARM). Users are organized within political hierarchies that define relative rights to access the resources. The DPCS system will distribute resources to groups of users by political priorities in accordance with established allocations and their recent usage. Under the constraints of political and other scheduling priorities, the DPCS system must be capable of considering the resource needs and requests of all jobs submitted to it, and of making an intelligent mapping of the job needs to the resources available.



The Offeror supplied CWARM system would be able to manage the various system components that comprise the entire environments, including, but not limited to, development, production, dedicated benchmarking, a mix of single-node jobs, a mix of multi-node parallel jobs, and jobs that use the entire system resource. This capability would be flexible enough to allow a rapid transition from one run-time environment to another. It would be able to configure run-time environments on an automated basis, such as by time and day of week. It would manage this transition in a graceful manner with no loss of jobs during the transition.

Production control of the DPCS through CWARM would span the entire system. That is, a job is an object that may be targeted to run on the entire system or a subset of the system. The resource management system would globally recognize a job throughout the system. A job may use 64b MPI libraries to span up to the complete system.

Jobs would be accounted for and managed as a single entity that included all its associated processes and memory. The DPCS through CWARM system would be able to dynamically collect and maintain complete information on the status of all the resources under its control at all times, so that the current pool of unused resources is known at all times.

It is anticipated that LLNL would port DPCS to utilize the CWARM supplied by the Offeror to quickly and reliably launch jobs, shepherd jobs through the system and accurately account for their system resource usage on an interval basis (not end of job accounting). The overhead of job management and accounting will necessarily need to be low in order to not destroy large job scalability (performance).

The ideal system would have reliability, availability, and serviceability (RAS) features integral to its design up to and including the full system. It would support such features as hot-swapping of components, graceful degradation, automatic fail-over, and predictive diagnostics. Supporting RAS features would be an on-site field engineering team that is staffed sufficiently to ably cover the site on a 24-hour 7-day basis, with two hour response during the prime shift by on-site personnel and a four-hour response time at night and on weekends and holidays by Q cleared personnel. The diagnostic tools the team employs would make possible the accurate diagnosis of problems to the field replaceable unit, thereby minimizing time-to-repair and repeated changing of parts hoping against all common sense that the next FRU replacement will be the actual failing unit. A sufficiently large on-site parts cache and hot-spare SMPs should be available to the on-site field engineering team so that SMPs can be quickly repaired or replaced and brought back on-line. A four hour return to production for down SMPs or other critical components during the day, and eight hours during off peak periods, is a strong requirement. A problem escalation procedure would be in place and would be invoked when necessary. Factory personnel would be available for solving particularly difficult problems. There would be a high degree of cooperation among the hardware engineers, field software analysts, LLNL personnel, and third-party suppliers. Engineering problems would be promptly reported to the appropriate engineering staff for immediate correction by an interim hardware release as well as in subsequent normal releases of the hardware. Appropriate testing and burn-in of the system components prior to delivery would also reduce the number of component “dead-on-arrival” and infant mortality problems.

Ideally the I/O subsystem would have a disk capacity of at least the baseline requirement, and would support RAID level 5. Due to the vast amount of storage required for Purple, an I/O subsystem with large MTBF characteristics should be architected. The system would have sufficient bandwidth to read and write in parallel large volumes of data, in two distinctly different usage patterns. Very large single files accesses by a large number of MPI tasks, one per CPU, in a non-overlapping fashion is one usage pattern. The second is one proportionally smaller file per MPI task, one per CPU, with all files in a single directory. This large number of files situation requires a fast file creation rate when a large number of MPI tasks open files from the same directory approximately contemporaneously. The I/O subsystem would also support a scalable parallel file system accessible from every node in the system. As the parallel file system is a critical system resource, it would be highly reliable.

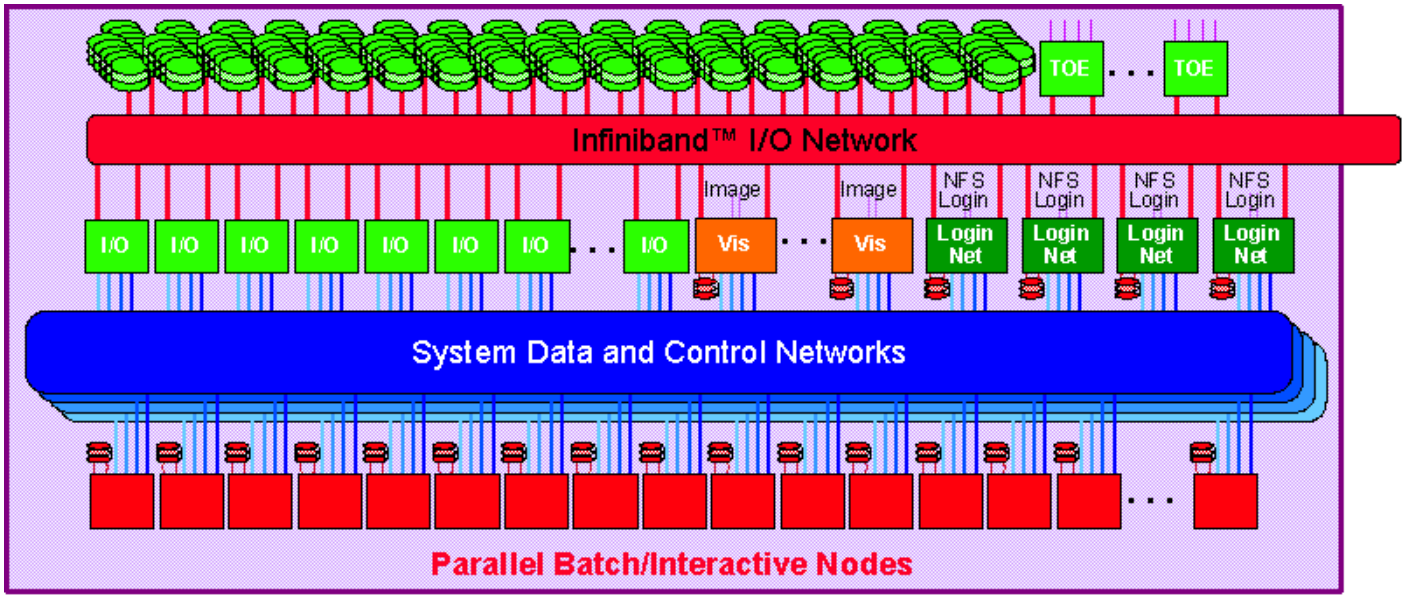


Figure 1.7-1 ASCI Purple SAN Architecture. Purple system architecture includes clustered I/O model, local node disks, dedicated login nodes , dedicated visualization nodes and compute nodes and Infiniband™ attached RAID disk and TOE I/O resources. Scalable user applications (either batch or interactive) will only run in the "parallel batch/interactive nodes" compute partition. The login nodes host interactive login sessions and code development activities, but not for running MPI jobs. The visualization nodes host parallel batch and interactive visualization jobs.

It is our intention to use Purple system as the vehicle for providing the first Storage Area Network (SAN) in to the LLNL classified computing infrastructure, see Figure 1.7-1. Therefore, it is essential that the I/O subsystem for connections for Purple be based on a SAN technology that can interoperate in this heterogeneous environment. At this time, it appears the leading contenders for this SAN technology are: Infiniband™, Fibre Channel, 1000Base-SW and 10GBase-SW. Our strategy is to accrete other systems (e.g., legacy SGI Origin 2000s visualization platform, Linux capacity clusters, distributed visualization clusters, and IBM SP based High Performance Storage System) to this SAN environment after Purple is integrated. The aim here is to have Purple provide scalable cluster wide parallel file system service for all of these resources over time. Thus the cluster wide file system (CWFS) supplied with Purple should be able to provide scalable global parallel performance to the other resources mentioned.

In addition, we expect TCP/IP off-load engines (TOEs) to be available for these competing SAN technology. These TOEs will allow extremely fast TCP/IP communications that don't burden the CPUs on the EDTV and Purple nodes originating the traffic. Thus the ideal Purple system will have outboard (to the Purple nodes) TOE devices that interface the SAN to the external networking environment.

External networking I/O to LAN, WAN, and SAN networks in the ideal system would support multiple protocols, perform channel striping, and have sufficient bandwidth to be in balance with the other elements of the system. Depending upon system protocol support, IP version 4 and IP version 6 traffic will be carried on the LAN and WAN. These circuits will support either IP over 1000Base-SW or 10 Gb/s Ethernet.

The operating software on the ideal system would be POSIX-compliant and provide a single-system image to both the programmer and the system administrator. However, this does NOT require that a single OS image or even a single SMP be employed. The system administration tools would allow for job scheduling across all SMPs in the system, allocation and partitioning of the system resources, user

accounts, and job queues in a variety of ways. The tools would provide ease of maintenance across all SMPs of system patches, configuration files, and addition and deletion of a new user. It would support backward code compatibility and retain the underlying programming paradigm. The system would maintain robust accounting files, system security files, and performance information. There would be a high degree of partnering on critical software elements. Similarly, in the ideal environment, a staff of Q-cleared, on-site software analysts would be available to support the system software. The University would have early access to new versions of the compilers, as well as other supplied software. Software releases for the supplier's products would be synchronized with operating system releases to ensure compatibility and interoperability.

The University requires some system resource availability in an unclassified mode of operation. It is required that the Early Deployment of Technology Vehicle (EDTV) system be split between classified and unclassified operations. When the final Purple system is delivered, the EDTV system would then be recombined on the unclassified network. In addition, to classified and unclassified operation, the University may require a second EDTV system for ASCI tri-laboratory collaboration.

The operating environment would conform to DOE security requirements. Software modifications would be made in a timely manner to meet changes to those security requirements.

1.8 Existing and New Terascale Simulation Facility (TSF)

An existing facility, the main computer floor of B451, will be used for siting the EDTV system. This facility has approximately 8,000-9,000 ft^2 and 1.9 MW of power for the computing system and peripherals and associated cooling available for this purpose in the "pop-out." Facilities modifications to provide power hook-up and cooling will be required to site the EDTV system. Thus, it is essential the Offeror make available to the University detailed and **accurate** (not grossly conservative over estimates) site requirements for the EDTV system (including the visualization system) at proposal submission time.

A new facility, known as the Terascale Simulation Facility (TSF), is scheduled to have the first computer room floor completed in the time frame of the proposed Purple system delivery. It will have approximately 190'x125' = 23,750 ft^2 of unobstructed computer floor space, will be able to accommodate up to 6 MW of power for the computing system and peripherals, and provide up to 2,000 tons of cooling capacity. This cooling, the total for the building in this time frame, can be used as chilled water or air cooling. All computer power is projected to be conditioned power, but not full UPS. Door opening dimensions and elevator and floor loading capacities will be appropriate for computer installations. A small temporary loading dock will allow for the delivery of equipment, and virtually no space will be available adjacent to the dock for equipment uncrating and staging. This area will lead directly to the computer room space. Uncrating and staging of equipment will have to be done on the computer floor. The computer floor is on the second floor and use of a 10,000 lbs. freight elevators will be required.

The initial phase of the TSF will be built with only one active computer floor. The second phase of the TSF project will make available approximately 288 offices. The third phase of the TSF project will make the second 23,750 ft^2 computer floor available. The second floor is currently planned to have 3 MW for power and 1,150 tons of cooling capacity. We would like to tailor the actual build of the first computer floor power downward to what the Purple system will actually require. Thus, it is essential the Offeror make available to the University detailed and **accurate** (not grossly conservative over estimates) site requirements for the Purple system (including the visualization system) at proposal submission time.

The University will be responsible for supplying the external elements of the power, cooling, and cable management systems.

The systems will be physically located inside an exclusion area within a security area. There will be no dialup capability to the classified portion of the system. No remote diagnostics will be allowed. Interaction of the on-site engineering staff with factory support personnel may be limited in some ways (e.g., dissemination of memory dumps from a classified system may not be allowed). This limitation emphasizes the importance of local access to source code, particularly for operating system daemons. Head disk assemblies (HDAs) from disks used for classified processing cannot be taken off-site or returned to the factory. All on-site personnel will require to be DOE Q-cleared or Q-clearable. It will be extremely difficult to provide access to foreign nationals.

On-site space will be provided for personnel and equipment storage.

A safety plan will be required for on-site personnel. They will be expected to practice safe work habits, especially in the areas of electrical, mechanical, and laser activities.

1.9 Purple Timescale and High Level Deliverables

Building and delivering a tera-scale computing resource at this scale is a daunting task. The successful completion of this project will require the very best efforts of the University and Offeror. It requires the careful planning and coordination of these efforts within the University and Offeror partnership. To this end, the University anticipates that the project will take on several critical stages: 1) formation of the partnership; 2) an Early Deployment of Technology Vehicle (EDTV) system for ASCI application code development; 3) demonstrate the technology refresh focusing on scalability of essential Purple hardware and software capabilities across the EDTV clustered SMP system; 4) demonstrate a peak plus sustained performance of at least eighty (80.0) teraFLOP/s on the two ASCI Purple Marquee application benchmarks; 5) Purple deployment to the program as the ASCI tri-laboratory capability platform; and 6) Purple deployment to the program as a general purpose production resource. If the Purple C option is exercised, then the peak plus sustained becomes at least one hundred thirty (130.0) teraFLOP/s. The table below gives general progress metrics for the successful completion of the Purple contract. These metrics include target dates based on ASCI programmatic requirements and anticipated fiscal year funding. These target dates are not mandatory and can be modified to more closely match an Offeror's product roadmap. In particular, the proposed delivery date of the Purple C option could be one to two quarters later than Purple in order to minimize cost and risk. However, there is a significant value to the University and the ASCI program to early delivery of technology and capability.

Target Date	Event	Metrics
July 2002	(1) Partnership Formation	Contract Signing and Development of initial overall project plan
September 2002	(2) EDTV	5 or 20 teraFLOP/s Peak Performance, 2.5 or 10 TB Main Memory, 100 or 400 TB RAID Disk. Five year EDTV maintenance clock starts after EDTV acceptance.
October 2002	(2) EDTV2 (Optional 2nd System)	5 or 20 teraFLOP/s Peak Performance, 2.5 or 10 TB Main Memory, 100 or 400 TB RAID Disk. Five year EDTV maintenance clock starts after EDTV acceptance.
3Q CY 2003	(3) Technology Refresh	Demonstration of key Purple hardware and software technology for applications scalability and system effectiveness on EDTV.

Target Date	Event	Metrics
4Q CY 2004	(4) Purple Demo and Delivery	Demonstration of Purple peak plus sustained on sPPM and UMT2000 of 80.0 teraFLOP/s performance. Delivery to LLNL. Memory to Peak FLOP/s ratio is 0.5. 1.2 PB RAID Disk
1Q CY 2005	(5) Purple Deployment	Purple stabilization ESP Rating of 80%. Start of limited availability. Start of five year maintenance clock.
1Q CY 2007	(6) Purple Production	Migration to heavy DSW workload and change in hardware/software maintenance. Start of general availability.

End of Section 1.0

2.0 High-Level Technical Requirements

The end product of the ASCI Purple development and engineering activity is a well balanced compute resource almost five times more powerful than those currently available at LLNL. It will be focused on solving the critical stockpile stewardship problems, that is, the large-scale application problems at the edge of our understanding of weapon physics. This fully functional Purple system must be useful in the sense of being able to deliver a large fraction of peak performance to a diverse scientific and engineering workload. It must also be useful in the sense that the code development and production environments are robust and facilitate the dynamic workload requirements.

The specifications below define a Purple system as a scalable SMP cluster with peak plus sustained performance of 80 teraFLOP/s (with a reasonable effort to achieve 87 teraFLOP/s) based on the performance of the sPPM and UMT2000 marquee demonstration codes identified in Section 9.1. The physics algorithms and coding styles of these two codes are indicative of key portions of the overall stockpile stewardship workload. Obviously, the Offeror will necessarily have to estimate the efficiency of sPPM and UMT2000 on the proposed system in order to determine what to actually bid, price and ultimately deliver to meet the mandatory requirement identified in Section 2.1.1.1. If the sum of delivered performance of sPPM and UMT2000 on the proposed system is below 20 teraFLOP/s, then more than sixty peak teraFLOP/s of computational resources will be required, and scaled as defined in 2.1.1.2. In any event, the sum of peak plus sPPM sustained plus UMT2000 sustained performance must be at least 80 teraFLOP/s (with a reasonable effort to achieve 87 teraFLOP/s). In the University's view, this issue will motivate additional Offeror innovation during contract execution.

Due to the classified and unclassified ASCI programmatic requirements the Purple system would be gainfully employed in the classified (RED) and the EDTV, after delivery of Purple would be gainfully employed in the unclassified (BLACK) network environments.

Development of the Purple shall comply with the requirements identified in section 8.0, Project Management.

There is only one mandatory requirement for Purple, section 2.1.1.1. The specific hardware and software highest priority requirements the Purple system may meet are delineated in sections 2.1 and 2.2, respectively, with (TR-1) designation. In addition to the highest priority hardware and software requirements, the Offeror will deliver any Target Requirements (TR-2 and TR-3) for the Purple, and any additional features consistent with the objectives of this project and Offeror's Research and Development Plan, which the Offeror believes will be of benefit to the project.

2.1 Purple Hardware High-Level Requirements

This section shall contain a detailed description of the proposed Purple System. It shall include a detailed discussion of how all of the Baseline Characteristics will be met, as well as a discussion of University and Offeror identified Value-Related Characteristics included in the technical solution.

2.1.1 Scalable Cluster of SMPs

2.1.1.1 Purple Scalable SMP Cluster (MR)

The Offeror shall provide a Purple teraFLOP/s system composed of multiple Shared memory Multi-Processors (SMPs) connected via a scalable intra-cluster communications technology. The

system shall have a peak performance of at least sixty teraFLOP/s (60.0×10^{12} floating point operations per second) and a peak plus sustained performance of at least eighty teraFLOP/s (80.0×10^{12} floating point operations per second) on the two SSP marquee benchmarks, sPPM and UMT2000. This is nominally 133% of the target peak.

2.1.1.2 Purple Component Scaling (TR-1)

In order to provide the maximum flexibility to the Offeror in meeting the goals of the ASCI project the exact configuration of the Purple SMP scalable cluster is not specified. Rather the Purple configuration is given in terms of lower bounds on component attributes relative to the peak performance of the proposed configuration. The Purple SMP scalable cluster configuration shall meet or exceed the following parameters:

- Memory Size (Byte/FLOP/s) ≥ 0.5
- Globally Addressable User Disk Space (Byte/FLOP/s) ≥ 20
- Memory Bandwidth (Byte/s/FLOP/s) ≥ 1
- Intra-Cluster Network Aggregate Link Bandwidth (Bytes/s/FLOP/s) ≥ 0.1
- Intra-Cluster Network Bi-Section Bandwidth (Bytes/s/FLOP/s) ≥ 0.05
- System Sustained Productive Disk I/O Bandwidth (Byte/s/FLOP/s) ≥ 0.001
- Cluster High Speed External Network Interfaces (bit/s/FLOP/s) ≥ 0.00125

The foregoing parameters will be computed as follows:

- Peak FP rate computation: Maximum number of floating point operations (chained multiply add counts as two) that can be completed per processor cycle on each processor times the cycle rate of the processors times the number of processors in the system. Peak FP rate is measured in teraFLOP/s = 10^{12} FLOP/s.
- Memory Size computation: Number of bytes of main memory directly addressable with a single LOAD/STORE instruction (but not caches nor ROM nor EPROM) of each node, summed over all nodes. Memory is measured in TebiByte (TiB) = 2^{40} Bytes.
- Globally Addressable User Disk Space computation: Number of user addressable bytes of RAID disk space. This does not include space required for RAID parity nor hot spares nor formatting nor file system overhead. The global disk space system must be accessible from every node of the system. Disk space is measured in TeraByte (TB) = 10^{12} Bytes.
- Memory Bandwidth/Peak FP (Byte/s/FLOP/s) computation: maximum number of bytes per second that some or all of the processors in a node can simultaneously move between main memory and processor registers (node memory bandwidth) times the total number of nodes divided by the peak FP of the systems.
- Intra-Cluster Network Aggregate Link Bandwidth computation: Intra-cluster network link bandwidth is peak speed at which user data can be moved bi-directionally to/from a an SMP over a single active network link. It is calculated by taking the MHz rating of the link time the width in bytes of that link minus the overhead associated with link error protection and addressing. The intra-cluster network aggregate link bandwidth is the sum over all active links in the system of the intra-cluster network link bandwidths. Passive standby network interfaces and links for failover can not be counted.
- Intra-Cluster Network Bi-Section Bandwidth computation: A bi-section of the cluster is any division of the nodes that evenly divides the total system into two equal partitions. A bi-section bandwidth is the peak number of bytes per second that could be moved bi-

directionally across the high speed interconnect network. The Intra Cluster Network Bi-Section Bandwidth is the minimum over all bi-sections of the bi-section bandwidths.

- System Sustained Productive Disk I/O Bandwidth computation: The system sustained productive disk I/O bandwidth is the measured rate an application can write productive data to the global disk arrays. The methodology for measuring this metric is specified in section 2.1.4.7. Note that this does not include delivered disk I/O bandwidth required for defensive I/O.
- Cluster High Speed External Network Interfaces (bit/s/FLOP/s) computation: The high speed external network interface link bandwidth (in b/s) is the HW rated link uni-directional bandwidth. This is the data rate, so it is 8 Gb/s for 10 Gb/s Ethernet. The cluster high speed external network interfaces bandwidth is the sum of over all the external network interface link bandwidths.

Example: For a 60 teraFLOP/s peak system, requirement 2.1.1.2 specifies that the system shall have at least 30 TiB of memory, 1.2 PB globally addressable user disk, 60 TB/s memory bandwidth, 6.0 TB/s intra-cluster network aggregate link bandwidth, 3.0 TB/s intra-cluster networking bi-sectional bandwidth, 60 GB/s system sustained productive disk I/O bandwidth and 75 Gb/s external networking.

2.1.1.3 Cluster Wide High Resolution Event Sequencing (TR-2)

The Purple shall include hardware support for a cluster-wide real-time clock or other hardware mechanism for cluster-wide event sequencing. The resolution of this mechanism shall be less than 1.0 micro-second ($1x10^{-6}$ seconds). This facility would be used for parallel program debugging and performance monitoring. The API overhead for obtaining the current clock reading from a user program on any node shall be less than five micro-seconds ($5x10^{-6}$ seconds).

2.1.1.4 Cluster Interconnect Reliability and Performance (TR-3)

If interconnect components fail (e.g., SMP network interface, network stage link, router or switch), each SMP will still be able to communicate with all other SMPs over the cluster interconnect. That is, there will be multiple interconnect paths or routes between SMPs so that, in the event of interconnect component failure, each SMP can still communicate with all other SMPs. The SMP interconnect will have the ability to segregate network traffic so that operating system related traffic (e.g., I/O service, OS to OS communication) does not interfere with user data communication traffic.

2.1.1.5 Cluster Interconnect Link Delivered Bandwidth (TR-1)

The sustained aggregate MPI message bandwidth available to/from each node shall be at least 80% uni-directional (i.e., 80% when sending/receiving messages of a size that optimizes system performance with MPI_SEND, MPI_RECV or MPI_ISEND, MPI_IRECV pairs) and 70% bi-directional (i.e., 70% with MPI_SENDRECV or MPI_ISENDRECV pairs) of the uni-directional and bi-directional aggregate link bandwidth, respectively. This will be measured as the minimum over all two-node pairs in the system with one MPI task per CPU in each node. A single MPI task communicating to one other MPI task on any other node in the system will have a sustained bandwidth of at least 50% of the delivered single processor memory bandwidth (i.e., 50% of processor memory bandwidth when sending/receiving a message of a size that optimizes system performance with MPI_SEND, MPI_RECV or MPI_ISEND, MPI_IRECV pairs).

Example: If the Offeror proposes nodes with 8 links with 2.0GB/s bi-directional bandwidth and a memory bandwidth of 3.0GB/s, then a job with one MPI task per CPU on the node will deliver 0.8*8*2.0/2 = 6.4 GB/s aggregate MPI_SEND/MPI_RECV bandwidth and 0.7*8*2.0 = 11.2 GB/s aggregate MPI_SENDRECV bandwidth. A single task on each of any two nodes in the system will deliver 0.25*3.0 = 0.75GB/s MPI_SEND or MPI_RECV bandwidth.

2.1.1.6 Cluster Interconnect Latency (TR-1)

The cluster interconnect latency, as measured by sending a minimum length MPI message from user program memory on one processor in the SMP cluster to user program memory on any other processor in the cluster and receiving back an acknowledgment divided by two (standard MPI user space ping-pong test), will be less than 5.0 micro-seconds ($5x10^{-6}$ seconds). When measured between any two nodes in the system with one MPI task per CPU on each node, the MPI user space ping-pong latency will be less than 10 micro-seconds (10^{-5} seconds).

2.1.1.7 Cluster Interconnect Delivered Bi-Section Bandwidth (TR-1)

The sustained aggregate MPI message bandwidth available to/from all nodes shall be at least 70% of the cluster interconnect bi-section bandwidth (i.e., 70% when sending/receiving messages of a size that optimizes system performance with MPI_SENDRECV or MPI_ISENDRECV pairs). This will be measured as the minimum delivered bandwidth over all possible combinations of communicating nodes. For each set of communicating nodes the aggregate delivered MPI bi-directional bandwidth is computed as the sum of delivered MPI bi-directional bandwidth over all two-node pairs in the system with one MPI task per CPU in each node.

Example: If the Offeror proposes a cluster with peak of 60 teraFLOP/s and a cluster interconnect bi-section bandwidth of 3 TB/s, then the delivered bi-section bandwidth will be greater than 0.7*3 TB/s = 2.1 TB/s.

2.1.1.8 Scalable Cluster Global Operations (TR-1)

The cluster interconnect and MPI library shall support scalable global operations under normal operating conditions. This will be measured by the GLOBAL_OP benchmark. That is, the GLOBAL_OP benchmark shall measure at least the following MPI global operations: MPI_Allreduduce, MPI_Reduce, MPI_Broadcast and MPI_Barrier. The MPI global operations shall be measured with the following methodology: for a given task count, iterate 10,000 times over doing one second of floating point work followed by the MPI global operation with one 64b floating point value and repeat. The runtime per iteration shall be measured as a function of MPI task count (NTASK) from 2 up to the number of CPUs in the system. The runtime per iteration as a function of NTASK shall remain constant or increase by at most the $\log_2$(NTASK). This benchmark shall be run under conditions matching those of the general workload (i.e., special calls requiring root access that perform task binding to CPUs or changing thread/process/task priorities is specifically not allowed) with normal system daemons running under normal operating conditions.

2.1.1.9 Remote Memory Access (TR-2)

The system will include support for remote memory reads and writes. A remote memory read returns the same data that a processor residing in the SMP with the remote memory location would have seen if it had read the location at the same time. Similarly, a remote memory write updates the location in a remote memory in the same way that a local processor memory write would have done. This remote memory access mechanism will include hardware memory

protection that protects the memory space of each individual user from all other users within the cluster. In order to hide latency, this mechanism will also support a large number of outstanding remote memory operations for each CPU in the node.

2.1.1.10 Additional Applications Memory for Cluster (MO)

As a separately priced option, the Offeror will install 15.0 TiB ($15x2^{40}$ Bytes) of additional memory in the Purple system utilizing the same density and performance memory components as configured in the base system. This option shall include sufficient additional disk to meet the swap space requirement defined in Requirement 2.1.4.4.

2.1.1.11 Additional Applications Memory for Compute SMP (MO)

As a separately priced option, the Offeror will install sufficient memory to double the memory in one Purple compute SMP utilizing the same density and performance memory components as configured in the base system. This option shall include sufficient additional disk to meet the swap space requirement defined in Requirement 2.1.4.4.

2.1.2 Shared Memory Multi-Processor

2.1.2.1 SMP Platform (TR-1)

The Shared memory Multi-Processor (SMP) platforms shall be a set of CPUs sharing random access memory within the same memory address space. The CPUs shall be connected via a high speed, extremely low latency mechanism to the set of hierarchical memory components. The memory hierarchy consists of at least processor registers, cache and memory. The cache will also be hierarchical. If there are multiple caches, they shall be kept coherent automatically by the hardware. The main memory may be a Non-Uniform Memory Access (NUMA) architecture. The access mechanism to every memory element shall be the same from every processor. More specifically, all memory operations shall be accomplished with load/store instructions issued by the CPU to move data to/from registers from/to the memory.

2.1.2.2 CPU Characteristics (TR-1)

Each SMP shall be an aggregate of homogeneous general purpose computers (CPUs) consisting of high-speed arithmetic, logic units, and memory, together with the necessary control circuitry and interprocessor communications mechanism(s). Each shall execute fixed and IEEE 754 floating-point arithmetic, logical, branching, index, and memory reference instructions. A 64-bit data word size shall directly handle IEEE 754 floating-point numbers whose range is at least 10^{-305} to 10^{+305} and whose precision is at least 14 decimal digits. The CPUs and memory hierarchy shall provide an appropriate mechanism for interprocessor communication, interrupt, and synchronization. The CPU will contain built in error detection and fault isolation for all CPU components and in particular for the floating-point units.

2.1.2.3 Minimum CPU Performance (TR-2)

The minimum single CPU performance, as measured by peak performance, will be at least four billion 64-bit floating point operations per second (4.0 GigaFLOP/s).

2.1.2.4 IEEE 754 32-Bit Floating Point Numbers (TR-3)

The CPUs will have the ability to operate on 32-bit IEEE 754 floating-point numbers whose range is at least 10^{-35} to 10^{+35} and whose precision is at least 6 decimal digits, for improved memory utilization and improved execution times.

2.1.2.5 Test-And-Set Instruction (TR-1)

The Offeror will provide sufficient atomic capabilities (e.g., test-and-set or load-and-clear) along with some atomic incrementing capabilities (e.g., test-and-add or fetch-and-increment/fetch-and-decrement) so that the usual higher level synchronizations (i.e., critical section, barrier, etc.) can be constructed. Additionally, these synchronization capabilities or their higher-level equivalents will be directly accessible from user programs.

The atomic instructions API overhead, in the absence of contention, shall be less than or equal to one micro-second (1.0×10^{-6} seconds).

2.1.2.6 Programmable Clock (TR-2)

There will be a real-time clock per CPU capable of causing a hardware interrupt after a preset interval (i.e., a programmable clock). The clock frequency will be at least one megahertz and the preset interval will be capable of being set in increments of 10 microseconds or less. There will be at least 16 seconds allocated for the time interval. This clock will have at least 24 bits.

2.1.2.7 Hardware Interrupt (TR-2)

The SMP will have hardware support for interrupting given subsets of computational processors based on conditions noted by the operating system or by other computational processors within the subset executing the same user application.

2.1.2.8 Hardware Performance Monitors (TR-1)

The CPUs will have hardware support for monitoring system performance. This published and well documented interface will be capable of separately counting at least the following: instructions (FP/INT/BR) per cycle with or without loads/stores; instruction cache misses; data cache misses; instruction TLB misses; data TLB misses; branch mispredictions; snoop requests; snoop hits; load miss penalty in cycles; number of pipeline flushing operations (e.g. sync). In addition, the SMP cluster interconnect interfaces will have hardware support for monitoring system performance. Memory hierarchy behavior, and message passing performance are of particular interest. This data will be made available directly to applications programmers and to code development tools (section 2.2.6.5.3).

2.1.2.9 Hardware Debugging Support (TR-1)

The CPUs will have hardware support for debugging of user applications, and in particular, hardware that enables setting data watch points (e.g., hardware interrupts on read/write to a specific virtual memory location). These hardware features will be made available directly to applications programmers in a published and supported API (section 2.2.6.5.3) and utilized by the code development tools including the debugger (sections 2.2.6.1, 2.2.6.1.2).

2.1.3 Memory Hierarchy

2.1.3.1 Shared Main Memory (TR-1)

The main memory (as distinct from cache memory) shall be high-speed, single-byte addressable, random access, single-bit correcting, double-bit detecting (SECDED). All components of the main memory (e.g., local and remote) shall be addressable by a single mechanism, (i.e., load/store instructions), from all compute CPUs in the SMP. Memory which is directly addressable by only a subgroup of the processors shall be part of a cache, if present. All caches within an SMP shall

be kept coherent automatically by the hardware. Note that this does not require that the distance (as measured in latency or bandwidth) to every memory element be the same from every processor.

Offeror will indicate additional recommended memory protection features such as RAIDed memory or other RAS features.

2.1.3.2 Node Shared Main User Memory Size (TR-1)

The node shared main user memory available to applications shall be larger than 22 GB. User memory available to applications shall be measured by a single MPI application with one MPI task on every CPU in the compute partition performing all to all communications as the amount of user data and code memory space available to the application. This specifically excludes memory required by the Operating System and buffers, required daemons (e.g., CWFS client) and other system services and MPI buffers.

2.1.3.3 Maximum Data Cache Size (TR-1)

Node memory hierarchies will include multiple levels of data cache. If the Offeror has memory hierarchy configurations that allow for selecting various data cache sizes, then the Offer will provide the maximum data cache size configuration.

2.1.3.4 Additional Physical Address Space (TR-3)

Each SMP will have the capability to directly address at least ten (10.0) TiB of physical memory with at least 46 bits of physical address space.

2.1.3.5 Memory Consistency Model (TR-3)

Within an SMP, a weak memory consistency model is allowed if there are specific hardware constructs (available to application programs) that can enforce a strong (sequential) memory consistency model for sequences of instructions (i.e., synchronization). This is sometimes known as a weak memory reference ordering model with additional hardware support for synchronization.

2.1.4 Input/Output Subsystem

The Input/Output subsystem for Purple has three major components: local disk, global disk and external networking. By the time Purple is delivered, Infiniband™ SAN and attached storage and networking solutions should be widely available. This solution is highly desired because it offers the opportunity to share disk and external networking resources between multiple platforms within the Livermore Computing High Performance SAN environment (e.g., capacity computing clusters, data manipulation engines, visualization engines, archival storage). The architectural picture Figure 1.7-1 shows the preferred system layout for the I/O subsystem.

2.1.4.1 Purple System Architecture (TR-1)

The Offeror will provide a concise description of the proposed Purple system and SAN architecture that includes the following:

1. System architectural diagram showing all nodes, networks, disks, external network connections and their proposed functions.
2. SAN architectural diagram showing all SAN networking components, connections to Purple I/O, visualization, login and TCP/IP Off-load Engines (TOE's).

3. Number of nodes required or recommended by the Offeror for system functions (e.g., cluster wide file system operation, switch operation and management, control workstation, stripe group managers, global lock managers) will be indicated and clearly denoted as NOT part of the parallel batch/interactive partition.
4. Describe each subsystem and it's system architectural requirements including bandwidths and latencies into, out of, and through each component. Clearly indicate the I/O performance limiters and bottlenecks.

2.1.4.2 Purple Clustered Wide File System and System Node Model (TR-1)

All global disk resources provided will be attached to the Offeror supplied SAN network and managed by dedicated file server nodes. The University's strong preference is for an Infiniband™ SAN network technology. However, if Infiniband™ is not part of the Offeror's product roadmap, then the Offeror may substitute Fibre Channel or GigaBit Ethernet. In the event that Fibre Channel or GigaBit Ethernet is the chosen SAN technology, then the SAN architecture should be reviewed by the University before proposal submission. Although these nodes will be included in the Peak teraFLOP/s and other system metrics (section 2.1.1.1) and utilized for the marquee demonstration runs, these nodes will not be part of the compute partition as fielded for testing, ESP determination nor production usage. If additional dedicated nodes (nodes unavailable for scalable user applications and hence not part of the parallel batch/interactive partition) and their fail-over backup are required for cluster wide file system operation or other system functions (e.g., distributed lock manager and backup, stripe group manager and backup, switch primary and backup), then the number of these nodes will be indicated in the system architecture and diagram.

2.1.4.3 Purple High-Availability RAID Arrays (TR-1)

All disk resources for the cluster wide file system will be RAID 3 or RAID 5 (or better) arrays. The RAID units will have high availability characteristics. These will include redundant fail-over power supplies and fans, at least one hot spare disk per eight RAID chains, hot swappable disks, the capability to run in degraded mode (one disk/RAID string failure), and the capability to rebuild a replaced disk on the fly with a delivered raw I/O performance impact of less than 30% on that RAID chain. There will be system diagnostics capable of monitoring the function of the RAID units and detecting disk or other component failure and monitoring read or write soft failures.

2.1.4.4 Purple Dual Boot Capability and Local Disk Space (TR-1)

Each SMP platform will have sufficient disk space for swap, local tmp space, NFSv4 cache and operating system, code development tools and other system binaries. This disk space, L, shall be the sum of the disk resources for each node (independent OS partitions on an SMP) L_i in an SMP.

$$L = \sum_{i=1}^{NP} L_i$$

Where, NP is the number of nodes on a given SMP.

Every node in each SMP platform will have the capability to boot two different versions of the operating system and all associated software (i.e., two completely separate and independent software releases or patch levels). Switching to a new boot device will be accomplished by the root user issuing commands at the shell prompt and will not require recabling any hardware.

The amount of node disk space will be at least four times the node memory (total for swap and local tmp space) in the node plus NFSv4 cache at 32 MB/CPU plus 1.5 times the sum of operating system, code development tools and other system binaries (sized at RFP submission time) per node of the SMP. All the swap and local tmp space will be used by each of the two dual boot environments. The other space will be duplicated for each of the two dual boot environments. In other words, the local disk size L_i (in GB) for node i, i=1, 2, … , NP is given by the formula:

$$L_i = 4*M + 2*[0.032*P_i + 1.5 * (O_i+C_i+T_i+G_i)]$$

Where, M_i = memory size of the node i in GB, P_i is the number of CPUs in node i, O_i is the size of the OS binaries, C_i is the size of the code development environment, T_i is the size of the tools binaries and G_i is the size of other binaries required for proper functioning of a single OS instantiation in GB.

Example: Suppose the proposed configuration has SMPs of size 128 CPUs with 256 GB of memory and four equal sized nodes (of 32 CPU and 64 GB of memory) per SMP. Suppose that the OS, code development environment, tools and other system binaries required 3.0 GB of disk space. Then, L = 4*4*64 + 2*(0.032*32 + 1.5*3.0) = 1,035 GB.

2.1.4.5 Single Process Sustained Serial I/O Bandwidth (TR-1)

The minimum sustained transfer rate for writing a single 50.0 GB file to a single logical file system will be no less than 100 MB/s. The file will be resident on either the local tmp or CWFS. That is, this requirement applies to both the local and cluster wide file systems. The benchmark for this metric will include the time to create the file, write 50.0 GB to the file (with block size chosen to maximize performance) and close and delete the file. In repeating this benchmark 100 times the runtime variance of all the runs will be less than 10%.

2.1.4.6 Multiple Sustained Serial I/O Bandwidth to Local Disk (TR-1)

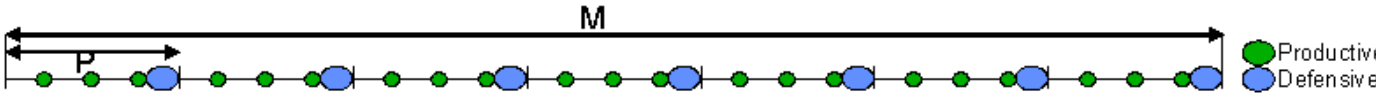
The minimum sustained transfer rate for every CPU in an SMP writing to a separate file on the node local tmp file system will be no less than *100*NP* MB/s, where *NP* is the number of nodes in an SMP (assuming more than one node per SMP with separate tmp per node) divided by the number of writers. The size of the files written will be MAX(50.0 GB divided by the number of writers, 3 GB). The benchmark for this metric will include the time to create the file, write the data to the file (with block size chosen to maximize performance) once, close and delete the file. In repeating this benchmark 100 times, the runtime variance of all the runs will be less than 10%.

Example: Suppose the proposed SMP has 24 CPUs and is proposed with 3 nodes (partitions), then the aggregate delivered local disk I/O bandwidth would be at least 100*3 = 300 MB/s. The delivered local disk I/O bandwidth per CPU (e.g., writer) will be at least 100/8 = 12.5 MB/s.

2.1.4.7 Parallel Sustained I/O Bandwidth (TR-1)

The amount of parallel I/O required for Purple depends heavily on the stability of the machine. ASCI simulations will take days to weeks to perform: much longer than any reasonable expectation of MTBAF (the mean time between application termination due to hardware failure). This is due to the fact that application restart dumps (essentially full process memory images minus the temporary data and code data for every process in the job) or checkpoint files need to be written much more frequently than the MTBAF. On some systems, it has been observed that 75% of the I/O to CWFS is in fact this type of defensive I/O. This type of I/O is not productive in the sense that typically only one or two restart dumps or checkpoint files needs to be kept around

at any point in time. Productive I/O on the other hand consists of graphics files, diagnostic physics data, key variable snapshots, etc. This is data that will be used by the scientist or engineer making the run to accomplish their mission requirements. It typically has a lifetime much longer than the run and needs to be moved to the archive, and the data analysis resources. Defensive I/O is only used for restarting a job in the event of application failure in order to not lose the state of the computation and hence the forward progress since the last application failure.*



Thus, the required amount of global I/O has two components: productive I/O and defensive I/O. The productive I/O requirement can be specified up front and does not require knowledge of the proposed MTBAF statistics. The defensive I/O requirement for Purple is expressed in terms of application MTBAF statistics. Suppose *M* is the MTBAF (in Hr) for a job utilizing 50% of real memory and 75% of the of processors in the compute partition. Suppose *S* is the time it takes to write a restart dump (in Hr). Suppose that *P* is the length of time between the start of successive restart dumps (in Hr). Averaging over many runs, the average amount of time lost due restarts is *P/2*. The amount of time that it takes write all the restarts is *S*M/P*. Thus, we wish to minimize:

(1) $T(P) = S * M / P + P / 2$

Hence, the *P* that minimizes *T(P)* is:

(2) $P = \sqrt{2 * S * M}$

The fraction of machine time lost due to having to do restarts is $f = T(P)/M$ or

(3) $f(M) = \sqrt{\frac{2 * S}{M}}$

Thus, if the memory size of the job is *m* (in TiB), the I/O rate (in TB/Hr) of the machine is given by:

(4) $R(M) = \frac{2 * m}{M f^2}$

or *R(M)* in GB/s:

(5) $R(M) = \left(\frac{10^3}{1800}\right)\left(\frac{m}{M f^2}\right)$

2.1.4.7.1 Parallel CWFS Defensive I/O Bandwidth (TR-1)

An application utilizing 75% of the CPUs in the Purple system compute partition and 50% of the real memory of the entire system will have fraction lost time due to system failure less than or equal to 10% ($f \le 0.10$). Thus, for a proposed system with peak performance (*F*, in teraFLOP/s) the defensive I/O rate (R_d, in GB/s) as a function of MTBAF (in hours) is given by:

$$R_d(M) \ge \left(\frac{10^3}{72}\right)\left(\frac{F}{M}\right).$$

* We are indebted to Michael Levine from Pittsburgh Supercomputer Center for the notion of defensive I/O and it's mathematical formulation.

The UMT2000 marquee code will be utilized to measure the delivered defensive I/O rate (R_d), actual MTBF, and f statistics over a 72 hour period as specified in section 2.1.4.7.3. Longer (shorter) MTBF ratings require less (more) defensive I/O.

Example: A 60 teraFLOP/s system with a 24 hour (one day) MTBAF for applications utilizing 25 TiB of memory and 85% of the processors in the compute partition yields a defensive I/O rate $R(24) \geq 35$ GB/s.

Example: A 60 teraFLOP/s system with a 504 hour (three week) MTBAF for applications utilizing 25 TiB of memory and 85% of the processors in the compute partition yields a defensive I/O rate $R(504) \geq 1.65$ GB/s.

2.1.4.7.2 CWFS Productive I/O Bandwidth (TR-1)

The Purple system will sustain productive disk I/O bandwidth (Byte/s/FLOP/s) of

$$R_p \geq 0.001 * F.$$

Where F is the peak of the Purple system. The IOR_MPI benchmark code will be utilized to measure this delivered productive I/O rate over a 24 hour period as specified in section 2.1.4.7.3.

Example: A 60 teraFLOP/s configuration will require 60 GB/s productive I/O rate.

2.1.4.7.3 Measuring CWFS Total I/O Bandwidth (TR-1)

To measure the cluster wide file system (CWFS) total I/O bandwidth, two tests will be utilized: UMT200 and IOR_MPI. Both applications will be running simultaneously over a twenty four hour period. UMT2000 will be set up to run on 75% of the processors in the compute partition and sufficient real memory so that each restart dump or set of restart dumps from a single checkpoint operation is at least 40% of the entire system memory size (i.e., $m=0.2*F$) for at least 24 hours of continuous runtime. UMT2000 will write application restart dumps to CWFS in order to minimize the lost time (i.e., no more frequently than specified by formula 2.1.4.7(2)). If UMT2000 aborts before the 24 hours of continuous runtime are completed, UMT2000 will be restarted from a previous checkpoint. The amount of UMT2000 runtime will be reset to when the last checkpoint ended. The amount of time lost between the completion of the previous checkpoint and UMT2000 successfully restarting shall be accumulated over the entire run. The total amount of time lost will be less than $f * 24$ hours, where f is determined by formula 2.1.4.7(3). UMT2000 will be configured for writing one file per MPI task using standard Fortran95 binary I/O calls under the following benchmark conditions:

1) UMT2000 will create a new directory under the checkpoint directory for this set of checkpoint files and cd to that directory.
2) UMT2000 will create one checkpoint file from each MPI task in the directory with zero length.
3) UMT2000 will write the checkpoint data to the file with blocksize chosen to optimize I/O performance.
4) UMT2000 will close the checkpoint files.
5) UMT2000 will cd to the checkpoint directory.

The wall clock time required for each MPI task ($j=1, NTASK$) to perform steps 1-5, above will be measured as (Tj). The checkpoint time for checkpoint dump k (T_k) is given by:

$$T_k = \underset{j=1}{\overset{NTASK}{MAX}}(T_j)$$

The defensive I/O rate for restart *k* (R_k) is defined as:

$$R_k = 0.25 * \mathrm{F}/T_K$$

Where F is the peak of the Purple system. The overall defensive I/O rate (R_d) is defined as

$$R_d = \underset{k=1}{\overset{N}{MIN}}(R_k)$$

Where N is the number of restart dumps completed in the 24 hour period.

While UMT2000 is running on 75% of the processors in the compute partition, IOR_MPI will run on 20% of the processors in the compute partition. Thus, at most 5% of the processors in the compute partition can be down while running this performance metric. IOR_MPI runtime will be much less than 24 hours and will be rerun every time it completes for the duration of the 24 hour period.

IOR_MPI will be configured for writing and then reading a parallel 40.0 TB file to the CWFS using MPI I/O calls under the following benchmarking conditions:

1) IOR_MPI will have one MPI task per processor.
2) IOR_MPI will create the file with zero size.
3) Each MPI task will write to non-overlapping partitions of the file where each partition size is optimized for performance and is contiguous space in the file.
4) After writing the file, each MPI task will read in a different non-overlapping, contiguous partition chosen at random.
5) IOR_MPI will verify that the data read in matches that written out.
6) IOR_MPI will close the file and delete it.

The time metric for this benchmark is the wall clock time (*W*) starting when the job is launched and stops when the job terminates minus the time it takes to perform step (5) above. Each run of IOR_MPI will execute steps 1-6 above 4 times. The IOR_MPI IO rate for run *k* (R_k) is determined by the formula $R_k = 2*4*40{,}000/W$. The overall productive I/O rate (R_p) is defined as

$$R_p = \underset{k=1}{\overset{N}{MIN}}(R_k)$$

Where N is the number of IOR_MPI runs completed in the 24 hour period.

2.1.4.8 Cluster High Speed External Network Interfaces (TR-1)

The Purple system will include a minimum of 0.00125 Bit/s per peak FLOP/s of aggregate external networking bandwidth. The Offeror will work with the University to assure these connections will be capable of interoperating with existing or future LLNL systems and network gateways.

These external network adapters will connect directly to the SAN network. The Offeror will configure no less than two and not more than eight login nodes to drive these external network adapters and to provide interactive login service to the system. These external adapters will off-load at least TCP/IP check summing and preferably the entire TCP/IP network stack for each connection.

The external network connections driven from the login nodes will deliver an aggregate of at least 65% of link bandwidth performance reading from the CWFS and writing to /dev/null on external network hosted sink resources.

Example: A 60 teraFLOP/s Purple system will have at least 0.00125*60,000,000 = 75 Gb/s external network bandwidth. This can be accomplished with approximately 8x 10Gb/s Ethernet (10,000 Base-SW) external network interfaces. This could also be accomplished with 7x 10Gb/s Ethernet (10,000 Base-SW) and 5x 1Gb/s Ethernet (1,000 Base-SW). The sustained FTP performance of 65% from CWFS to externally hosted /dev/null resources will yield more than 0.65*75 Gb/s = 48.75 Gb/s ≈ 6.1 GB/s. To reach this level of performance, 10,000 Base-SW and 1,000Base-SW Jumbo Frame support will be required.

2.1.4.9 Additional Global Disk (MO)

As a separately priced option, Offeror shall bid 25% additional global disk that is fully configured into the CWFS.

2.1.5 Early Access to Purple Hardware Technology (TR-1)

The Offeror may propose mechanisms to provide the University early access to Purple hardware technology for hardware testing that includes other steps before inserting the technology into EDTV. Small additional early access systems are encouraged.

2.2 Purple Software High-Level Requirements

2.2.1 Operating System

2.2.1.1 SMP Base Operating System and License (TR-1)

The Offeror will provide a standard multiuser Single UNIX Specification Version 3 or later and the UNIX 98 Server Product or later branded operating system (http://www.unix-systems.org/version2/unix98.html). Alternatively, the Offeror will provide a standard multiuser Linux standards base specification V1.0 or later compliant interactive operating system (http://www.linuxbase.org/spec/). The base operating system will consist of at least a basic kernel that supports system services and multiprocessing applications. Fully supported kernel-level implementation, as defined by the POSIX 1003.4 (or later) working group standard of thread operations in shared address spaces shall also be provided (within six months of standardization or at Purple delivery). The operating system shall provide mechanisms to share memory between user processes and to run threads within a single user process on multiple CPUs simultaneously. This shall include provision of right-to-use license for an unlimited number of users, including unlimited concurrent usage, of the operating system, daemons, and associated utilities. The University will accept the Offeror's self-certification for POSIX compliance.

2.2.1.1.1 UNIX/Linux OS Compliance (TR-2)

The proposed operating system will have the Single UNIX Specification Version 3 or later and the UNIX 98 Server Product or later brand. Software with the functionality of the following Single UNIX Specification components will be provided:

(C604) Commands and Utilities, Issue 5;
(C605) System Interface Definitions, Issue 5
(C606) System Interfaces and Headers, Issue 5

(C610) X/Open Curses, Issue 4, Version 2
(C808) Networking Services (XNS), Issue 5.2

For the Single UNIX Specification Version 3 or later and the UNIX 98 Server Product or later brand, the Offeror will deliver a copy of the Single UNIX Specification Version 3 or later and the UNIX 98 Server Product or later brand with the Purple delivery.

2.2.1.2 Single System Image (TR-2)

The Purple SMP scalable operating system will present a single system image to users. The single system image will include: single cluster login address; cluster wide load leveling at login and at process or job creation time; cluster wide file lock management; cluster wide process and POSIX session ID space; virtual memory page sharing between SMPs; and extensions of standard UNIX utilities to the single system image.

2.2.1.3 Thread Packages (TR-2)

If the Offeror bids UNIX (as opposed to Linux), then the Offeror will provide an implementation of the IEEE POSIX 1003-1996 (.1c) threads and M:N thread support. A user application will operate correctly when using the combination of: manual POSIX threads, and compiler generated threads (requirement 2.2.5.1 and section 2.2.5.9). That is the POSIX threads, and compiler generated threads will interoperate in a single user application binary, as long as the POSIX threads are at the same level as the compiler generated threads.

2.2.1.4 SMP Partitioning (TR-3)

If the Offeror proposes to partition the SMPs into multiple nodes with independent OS images on each node, then the OS will support the partitioning of the SMPs so that user applications can still access memory between nodes through the MPI library. In addition, booting of individual nodes will be independent. Each node will be able to have different versions or patch levels of the OS and other supplied software running. MPI interoperability need only be supplied for nodes with the same OS and other supplied software at the same version and patch level. Describe any additional features provided by SMP partitioning.

2.2.1.5 Dual Boot Capability (TR-1)

The node operating system will have the ability to boot from at least two different environments. Switching between the two boot environments will be accomplished by the root user issuing commands from the shell prompt and rebooting the node. No manual hardware reconfiguring will be required to switch boot environments. Once running a boot environment, it will be possible to apply system installs, patches and configuration changes to both the active and the non-active boot environment. The supplied operating system will share (reuse) the swap and local /tmp space in each of the two boot environments. Other required file systems (e.g., /) will be replicated.

2.2.1.6 Pluggable Authentication Mechanism (TR-1)

The Offeror will provide a service programming interface (SPI) that allows the replacement of the standard authentication mechanism with a University provided pluggable authentication mechanism (PAM). The SPI shall be supported by all Offeror supplied login utilities and authentication APIs. The purpose of this PAM is to allow the University to meet changes in DCE security requirements and LLNL to implement stronger authentication (e.g., one-time password authentication).

2.2.1.7 System Utilities (TR-2)

The Offeror will provide the GNU Utilities, which will at least include: top, bash, emacs, make; a scripting language such as perl; and lint-like tools for the baseline languages.

2.2.1.8 Fully Integrated System Initiated Parallel Job Checkpoint/Restart (TR-2)

The system software will support a mechanism to save to disk the current state of a parallel job (utilizing any combination MPI and SMP parallelization) running across the entire Purple system (checkpointing) and later restore this application to a running state (restart). The portion of the parallel job resident on individual nodes or SMP platforms will be written/read from memory/local disk to local disk/memory independently and contemporaneously. That is, the checkpoint/restart will be accomplished in parallel. This mechanism will require no special system calls nor any special linking requirements in the application program being checkpointed.

2.2.1.8.1 Checkpoint/Restart Minimum I/O Rate (TR-2)

A parallel job utilizing 15.0 TB of system memory will be checkpointed/restarted to/from local disk or CWFS in less than 300 seconds (50.0 GB/s delivered overall).

2.2.1.8.2 Operating System Support for Parallel Job Checkpoint/Restart (TR-2)

The Offeror supplied operating system will support system initiated checkpoint/restart of SMP and MPI parallel jobs. The operating system software will also support a mechanism by which the job management system can initiate the storage of the complete state of a parallel job to "checkpoint files." This mechanism will not require any special system calls or any special linking requirements in the application program being checkpointed. The system software will support resumption of a parallel job from the point at which it was checkpointed using the data in these checkpoint files. The operating system software will support a mechanism by which data for an active job can be moved out of memory in a single "roll-out" operation (rather than by demand-paging) when the job is descheduled. The operating system software will support a mechanism by which swapped-out data for a descheduled job can be moved back into memory in a single "roll-in" operation (rather than by demand-paging) when the job is rescheduled. It will be possible to move checkpoint files to a different node or SMP platform on the same system and to restart the job from the point at which it was checkpointed.

2.2.1.8.3 Minimal Job Restrictions For Checkpoint/Restart (TR-2)

The Offeror supplied operating system will be able to reliably checkpoint/restart the ASCI applications workload. That is, the set of checkpoint/restart restrictions on jobs will allow jobs with the following characteristics to be checkpoint/restartable:

1. Shared memory between threads and processes of a job
2. Open files for append with writes in progress
3. MPI messages in flight

2.2.1.8.4 Job Notification Before A Checkpoint Operation (TR-2)

The Offeror supplied operating system will notify a job via a published mechanism before it is checkpointed so that job can clean up (become conformal with job checkpoint/restart restrictions) and prepare to be checkpointed. The time interval between sending the checkpoint signal and the actual checkpoint operation will be a system wide configuration parameter.

2.2.1.8.5 User Selectable Checkpoint File Destination (TR-2)

The Offeror supplied operating system will allow the user to specify the directory where checkpoint files will be written with an published environment variable or published API. A default location will be supplied that allows the user to create/write/read/delete privileges. The system will not allow the default or user specified checkpoint file location to be other than local disk or the CWFS. In particular, the system will not allow NFSv3 or NFSv4 mounted file systems as destinations for checkpoint files.

2.2.1.8.6 Fault Tolerance for Checkpoint Files on Local Disks (TR-2)

If the Offeror supplied operating system allows checkpoint files to be written to node local disk resources, then the checkpoint facility will be able to restart from these checkpoint files if as many as one in eight local file systems are not available a the time of job restart. Note that this may be accomplished via software RAIDing the checkpoint files across multiple local devices.

2.2.1.9 Cluster Wide Fault Tolerance and Graceful Degradation of Service (TR-2)

The operating system will have the ability to detect, isolate and manage hardware or software faults in a way that minimizes the impact on overall SMP cluster availability. When SMP cluster (hardware or software) components fail, the SMP cluster software resources will provide degraded system availability. Under most circumstances, it will be possible to take hardware and software components off-line or bring them back on-line without operating system rebooting. The probability that a job will fail (due to hardware or software faults) should be proportional to the amount of resources consumed by the job, not SMP cluster size.

2.2.1.10 Networking Protocols (TR-1)

The operating system shall support the Open Group (C808) Networking Services (XNS) Issue 5.2 (http://www.opengroup.org/pubs/catalog/c808.htm), or later, standard networking protocol suite over the network interfaces described in requirement 2.1.4.8. In particular, over these interfaces the IPv6, TCP/IP, UDP, NIS, NFS (client and server), RIP, telnet, ssh, and ftp protocols shall be supported.

2.2.1.11 Disk Sharing Between CWFS and HPSS (TR-2)

It is anticipated that High Performance Storage System (HPSS) disk cache will be shared with Purple global file system. That is, part of the Purple CWFS disk will be utilized by HPSS as disk cache. The Offeror will provide the appropriate networking drivers, protocol support and file system hooks to enable the transfer of control of files in the Purple file system to HPSS. Once files are migrated by HPSS to tape, the disk storage resources associated with the file will be released back to the Purple file system for it's use. The Offeror will provide the appropriate networking drivers, protocol support and file system hooks to enable the dynamic reacquisition of disk resources no longer needed by migrated files back to the Purple file system from HPSS.

2.2.1.12 Local File Systems (TR-2)

The operating system local file system will have a POSIX interface that is 64b by default and will support individual files of at least ten (10.0) GB in size. The local file system will support individual file systems of at least two (2.0) TB in size. The file systems will support increased reliability and fast reboots (e.g., reduce the FSCK time via a journal implementation). That is, the file system will be designed and implemented so that any file system initialization that delays system reboots or file system restarts/mounts will have at most logarithmic complexity in the

number of devices and files/directories. The aggregate file system initialization and file system restarts/mounts shall be less than five (5.0) minutes for all the proposed node local file systems. The local file system will have a logical volume manager that allows the striping of all file local file systems (including the root or /, /swap, /usr and /var) across multiple disks in order to maximize performance. The logical volume manager will be able to migrate directory structures and associated files to different physical devices and add/subtract disk blocks to a file system. The local file system shall support dual boot capability (section 2.1.4.4 and 2.2.1.4) by being able to mount all the partitions of the other boot environment. The provided file system and logical volume manager will have a utility that will scan the file system metadata and data disk blocks and repair damage to the file system while the file system is mounted for normal usage.

2.2.1.13 Cluster Wide File System (TR-1)

The operating system file system will support a cluster wide file system (CWFS) with a POSIX interface that is 64b by default and will support individual files of at least ten (10.0) TB in size. The cluster wide file system will support individual file systems of at least two (2.0) PB in size. That is, the file system will be designed and implemented so that any file system initialization that delays system reboots or file system restarts/mounts will have at most logarithmic complexity in the number of devices and files/directories. The aggregate file system initialization and file system restarts/mounts shall be less than one (1.0) hour for all the CWFSs. The CWFS will be configurable to utilize all of the global disk (section 2.1.1.2) in a single file system. In other words, CWFS will support the number of RAID devices proposed for the I/O subsystem in a single file system. CWFS and subsystems will have a utility that will scan the file system metadata and data disk blocks and repair damage to the file system while the file system is mounted for normal usage. The CWFS will be capable of mounting in the event of one or more (up to 10% of the) RAID devices are not functioning. In the event of unrecoverable RAID device failure, the CWFS will have a supported and documented utility that can be run by the system administrators that will scan the file system metadata and create a listing of all the corrupted directories and files due to the lost RAID device, sorted by user, in less than two (2.0) hours. The CWFS will be configured so that if a single CWFS server fails another server will take over servicing requests for I/O to the devices owned by the failing server. The degradation in performance to the CWFS will be less than 20% when running in degraded mode with one failed server. The system diagnostics will notify the operators that the fail over has occurred within ten (10) minutes of fail over.

2.2.2 Distributed Computing Middleware

2.2.2.1 OSF DCE (TR-1)

The Offeror shall provide the Open Software Foundation (OSF) Distributed Computing Environment (DCE), version 1.2.2 or later, client software on the proposed cluster of SMPs. This shall include a fully supported DCE integrated login mechanism that will support the use of password authentication and Kerberos V5 ticket authentication against a DCE security server. This mechanism shall comply with the authorization policies established for DCE accounts (e.g., password lifetime, account lockout) and shall be capable of acquiring and storing the user's DCE credentials for a login session. This mechanism along with the following login utilities and daemons- rsh, rcp, rlogin, telnet, ssh and ftp, shall be fully interoperable with the login utilities and daemons in Kerberos V5.1.2, or later, as distributed by MIT or in Secure Shell V2 distributions for ssh.

2.2.2.2 NFSv4 Server for Exporting CWFS (TR-1)

The cluster wide file system (CWFS) will be exportable via the NFS version 4 (NFSv4) protocol. It will support the use of RPCSEC_GSS for authentication, integrity and privacy and will include support for, but should not be limited to, Kerberos V5 as a security mechanism. In addition, the CWFS will provide the NFSv4 Access Control List (ACL) mechanism for directory and file access control. The ACL mechanism shall provide support for users and groups in a multi-domain environment (i.e., recognize identity@domain identifiers). Access checks or representation of file or directory ownership or group ownership in a NFSv4 exported CWFS file system shall recognize the RPCSEC_GSS authenticated identity. Evaluation of ACL entries will adhere to the NFS version 4 protocol (IETF RFC3010)..

2.2.2.3 NFSv4 Client (TR-1)

The Offeror shall provide NFS version 4 client software on each node in the proposed cluster of SMPs. This shall include the use of RPCSEC_GSS for authentication, integrity and privacy with support for, but should not be limited to, Kerberos V5 as a security mechanism. This shall include a fully supported NFSv4 ACL mechanism with ACL editing utilities. The ACL mechanism shall provide support for users and groups in a multi-domain environment (i.e., recognize identity@domain identifiers).

2.2.2.4 Cluster Wide Service Security (TR-1)

All cluster wide services including debugging, performance monitoring, event tracing, resource management and control will be performed using a secure authentication and authorization protocol that interfaces to the PAM (section 2.2.1.6).

2.2.3 Cluster Wide Accounting and Resource Management (CWARM)

2.2.3.1 Job definition (TR-1)

A Job is defined as a cluster wide abstraction similar to a POSIX session, with certain characteristics and attributes. The characteristics and attributes required for each job type are as follows:

1. Interactive Job: An interactive job includes all cluster wide processes and threads executed as a child (whether direct or indirect through other processes) of the command invoked from within an interactive login to initiate the parallel or remote execution of the job (e.g., MPIRUN). Interactive jobs will run with any combination of both MPI and SMP parallelism.
2. Batch Job: a Batch Job includes all cluster wide processes and threads executed as a child (whether direct or indirect through other processes) of a shell process executed as a child process of a batch system shepherd process, and includes the batch system shepherd process as well. This includes all processes and threads spawned as a result of the command executed to initiate parallel or remote execution of the job from a batch script. Batch jobs will run with any combination of both MPI and SMP parallelism.
3. Ftp Job: an ftp job shall include an ftpd and all its child processes.
4. Kernel Job: all processes with a pid of 0.
5. Idle job: this job does not necessarily actually consist of identifiable processes. It is a pseudo-job used to report the lack of use of resources.
6. System session: all processes owned by root that are not a part of any other job.

2.2.3.2 Minimal Resource Set (TR-1)

The minimal set of resources managed through the use of Offeror utilities and GUIs and published APIs shall include:

1. number and type of nodes;
2. number of CPUs;
3. per-CPU and total CPU time;
4. wall clock time;
5. real memory limit (high-water allocation);
6. real memory integral;
7. virtual memory integral;

2.2.3.3 Cluster Wide Job Management (TR-1)

Commands will be available to reliably manipulate a job as a single entity (including kill, modify, query characteristics, and query state).

2.2.3.4 Cluster Wide Resource Management (TR-2)

The Offeror will supply an X11-based graphical user interface for a single point of control resource manager to manage a minimal set of resources which includes number and type of CPUs, per-CPU and aggregate CPU time, wallclock time, memory (high-water allocation), and temporary disk space, as well as more detailed information such as CPU usage by user, network usage by hostname, and memory integral and swap memory integral.

2.2.3.4.1 Cluster Wide Resource Limits (TR-2)

The Offeror will provide a capability to assign and detect/report a soft limit for each supported resource and enforcement of a hard limit for each supported resource, as defined requirement 2.2.3.2.

2.2.3.4.2 Cluster Wide Batch Resource Management API (TR-2)

The Offeror will provide a published batch scheduler API for getting the configuration and status of individual nodes as well as determining the configuration and status of overall system resources. The batch scheduler API will provide a function to submit and terminate single-threaded and multi-threaded MPI jobs. All batch scheduler APIs shall return documented error status codes and shall not write error logs to STDOUT/STDERR. The batch scheduler API will provide a function to report CPU, virtual memory, real memory and paging load metrics.

2.2.3.4.3 SMP Job Geometry Specification (TR-2)

The batch scheduler job command language will permit the specification of job node requirements and MPI task and OpenMP thread layout within nodes. The interactive parallel job launch interface will also permit the specification of job node requirements and MPI task and OpenMP thread layout within nodes through command line options. Using either method (batch or interactive) to start a parallel job, users shall be able to specify any logically possible MPI task and OpenMP thread layout and packing order. The job launch facility will ignore geometry specification options if executed within a Batch Job script.

2.2.3.5 Cluster Wide Job Accounting (TR-2)

The Offeror will provide published GETPROC() and GETUSERINFO(), or equivalent, APIs to obtain all process data on the node of the cluster the calls are issued on. The data returned shall be an up-to-date snapshot of resource usage as of the time of the call to the API routine.

The Offeror will provide inetd that makes the operating system call "setsid()" in addition to the typical "setuid()" and "initgroups()" calls before exec'ing subsystem processes.

The Offeror will provide a published low overhead (because it will be called often) API callable on any node that returns a list of master processes running on that node. The Offeror shall provide a published API callable on the host node that, given a process ID that spawned a parallel job, returns an array of structures with a structure for every spawned process. At a minimum each structure in the array shall contain: 1) node name; 2) Unix/Linux session ID; 3) process name; and 4) PID.

2.2.3.6 Cluster Wide Job Scheduling (TR-2)

The batch and interactive job launch facility will provide a mechanism that a non-root user can use to submit a job that spans any subset of user accessible processors, and to schedule multi-thread, multi-process, message-passing jobs using configurable load levels. These configurable load levels shall be defined on CPU load, available memory, paging rate (if applicable) and available swap. The Cluster Wide Job Scheduling (CWJS) environment will fully support parallel system initiated checkpoint/restart of all jobs in the system. The CWJS workload suspend command will checkpoint the entire running workload for controlled system shutdown. The CWJS workload resume command will restart the system workload from a controlled shutdown. Jobs in batch run queues will survive the system workload suspension and resumption. The CWJS system will keep system utilization and throughput statistics. The CWJS shall externalize, via a documented API, the checkpoint/restart of a single parallel job command so that ASCI distributed resource management (or other higher level resource management) system can utilize this API for guaranteeing the delivery of system allocations.

2.2.3.6.1 Fast, Scalable and Reliable Job Launch (TR-1)

The CWJS will utilize a fast, scalable, reliable mechanism to launch parallel jobs. The job launch mechanism is scalable if the job launch time does not depend on the number of nodes the job is launched on (i.e., job launch utilizes a switch hardware broadcast mechanism for binary distribution). Job launch is reliable if job launch fails less than one in one thousand job launch requests integrated over the entire ASCI workload. Job launch is fast if it can launch UMT2000 set up to measure (requirement 2.1.4.7.3) defensive I/O rates and the "date" command on every CPU in the compute partition in under five (5) minutes and the variation of job launch time is under ten percent (10%).

2.2.3.6.2 Cluster and SMP Gang Scheduling (TR-2)

Offeror will provide a cluster and node Gang scheduling functionality. The Offeror will provide a robust capability to gang schedule threads and processes from a single user job within a node and across nodes and SMPs. That is, when a user job is scheduled to run, the gang scheduler shall contemporaneously allocate to CPUs all the threads and processes within that job (either within a node or within the cluster of SMPs). This scheduling capability shall control all threads and processes within the cluster. The GANG scheduling mechanism will timeshare and spaceshare nodes and SMPs between multiple parallel jobs by suspending jobs

that are not scheduled and resuming jobs that are scheduled to run. The GANG scheduling mechanism shall be externalized via a published API.

2.2.3.6.3 Gang Scheduler Job Suspend/Resume (TR-2)

The gang scheduler job suspend/resume functionality will permit the automatic and manual suspension by the resource manager or root user and resumption of all jobs running in the system. Normal users will not be able to cause the suspension or resumption of their jobs. The gang scheduler will suspend jobs "in place" when the resident set size of all jobs that overlap with that job fit into the available real memory of the nodes running those jobs. When the sum of the resident set sizes of jobs running on a node exceed the available real memory of the node, then the gang scheduler will utilize checkpoint/restart to the CWFS to suspend/resume those jobs.

2.2.3.6.4 Gang Scheduler Dynamic Configuration (TR-2)

The resource manager will have the ability to statically set via configuration files and dynamically modify the configuration of the gang scheduling mechanism. This includes, at a minimum, the time slice parameter (i.e., minimum interval between job resume and job suspend). After modification, the change will take effect without having to restart the gang scheduler, resource manager or reboot the Purple system or any components.

2.2.3.6.5 Gang Scheduler and Debugger Integration (TR-3)

The gang scheduler will be fully integrated with the debugger. This integration will include the ability for the gang scheduler to provide fast response time to jobs launched interactively under the control of the debugger by suspending other running jobs to create room for the job being debugged. When the debugger hits a break point, the gang scheduler will have the ability to suspend the job and allow other jobs to run until the user continues or steps the job at which time the gang scheduler will have the ability to resume the job. Debugger control of the job (e.g., viewing or changing program variables under the control of the debugger, setting or clearing break points, etc.) will not be impeded by the gang scheduler.

2.2.3.6.6 Dynamic Job Migration (TR-3)

The CWJS will optimize job placement across the SMP cluster by utilizing parallel system initiated checkpointing of jobs, movement of the checkpoint files to a new set of nodes and restarting the jobs, if doing so would improve system throughput and/or responsiveness.

2.2.3.6.7 Cluster Wide Job Scheduling Interface (TR-2)

The Offeror will provide an X11-based graphical user interface using an X window display which displays machine loads and systems information (including as a minimum the items described in requirement 2.2.3.1) for batch and interactive scheduling classes. This interface shall also provide historical usage, job tracking from submission to deletion (note that deletion is not the same as end of execution), capability to control distribution of resources between interactive and batch requests, ability to tailor the scheduling policy to specific site requirements, and the ability to manually adjust cluster-wide load balance by checkpoint/migration/restart of single process jobs. In addition, job status shall be retained at least 24 hours after termination. All of the job scheduling functions underlying this X11-based user interface shall also be provided in a published API.

2.2.3.6.8 Operator Interface Support for Parallel Job Checkpoint/Restart

The resource management control interface will fully support parallel system initiated checkpoint/restart of all jobs in the system. This interface shall show active and checkpointed jobs. This interface will allow system operators to initiate the suspension of the running workload and resumption of a suspended workload as a single command. This interface will allow the operator to tune GANG scheduling parameters (such as slice time) without needing to restart the workload nor the operating system nor batch or interactive job launch software subsystems. This interface shall present the operator with system utilization and throughput statistics.

2.2.3.7 Cluster Wide Job Accounting (TR-2)

The Offeror will provide job accounting, with API interface, where data for all threads and processes of a job are combined to provide an aggregate job accounting record, and individual thread and process records are identified as being related to a job, for at least the minimal resource set. The accounting information will also contain UNIX UID information. The data returned will be an up-to-date snapshot of resource usage as of the time of the call to the API routine.

2.2.3.8 Accounting for System (Root) Usage (TR-2)

Resources used by root processes that are not otherwise considered to be part of another job and by the operating system itself will be accountable. In addition, resources not used will be accountable. Accounting for these resources will be accomplished through the same interfaces as are provided for all other jobs. Accounting for operating system resource usage will be done through a pseudo-job known as the "kernel" job. Accounting for root processes' resource usage will be done through one or more pseudo-jobs known as "system" jobs. Finally, accounting for unused resources will be done through a pseudo-job known as the "idle" job.

2.2.4 Cluster System Administration Tools

2.2.4.1 Single Point for Cluster System Administration (TR-1)

The Offeror will provide a set of facilities to administer the Purple SMP cluster as a single entity. In particular, the Offeror shall provide fully supported implementation of a single-point system administration tool to effect configuration actions on: file system mounts; node or SMP booting; node or SMP status; node or SMP self-consistency checks of system configuration parameters; software installation; resource administration; node or SMP shutdown/restart; system patch installation; login control (provide capability to restrict login access to certain processors, and cluster-wide monitoring of failed login attempts by an individual); and system back-ups, including ability to dump multiple volumes of tapes without operator intervention. This single-point of control shall provide a command-line interface that effects one or more actions from each command issued with error return code allowing the system administrator the ability to script (automate) redundant configuration tasks for multiple or all SMPs or nodes in the cluster. This command-line interface shall be capable of performing all of the above system administration configuration actions. The Offeror will provide a fully supported implementation of mechanisms for detecting and reporting failures of critical resources, including processors, network paths, and disks. The diagnostic routines shall be capable of isolating hardware problems down to the FRU level in both the system and its peripheral equipment.

2.2.4.1.1 Fast, Reliable System Reboot (TR-1)

Rebooting the entire Purple system will take less than one (1) hour and once initiated will not require human intervention. This time will include the time to reboot the nodes, switches and mount all file systems and return all system daemons (including the CWFS) to operating condition.

2.2.4.1.2 Fast Software Installation and Reversion (TR-1)

Software installation or upgrade of the entire system will complete in eight (8) hours when the entire system is in a dedicated mode and unavailable to users. Reverting to a previous software release shall complete in four (4) hours. This includes any system reboots. Single node or SMP installation will complete in less than one (1) hour.

2.2.4.1.3 Fast Software Patch Installation (TR-1)

The installation of bug fixes and minor software enhancements (patches) will take less than two (2) hours, not including rebooting, when the entire system is in a dedicated mode and unavailable to users. It will be possible to back out patches. It will take less than two (2) hour, not including reboot, to back out a patch.

2.2.4.1.4 Alternate Configuration Boot, Install and Patch (TR-1)

The system will have the ability to boot two alternate system software release and/or configurations. This will be used to test new system releases in "debug shots." Switching between these two will be accomplished with a single system reboot and take less than ninety (90) minutes. Installing, upgrading, and patching the entire alternate configuration (the configuration that is not live) will be accomplished with the system on-line and under user workload and will take less than twenty four (24) hours for installs and upgrades, and two (2) hours for patches.

2.2.4.2 Cluster System Debugging and Performance Analysis (TR-2)

The Offeror will provide a set of facilities with a single-point of control to analyze the entire Purple system performance and make tuning modifications. In particular, the Offeror will provide fully supported implementation of a single-point of control system tuning tool to dynamically monitor and modify the following system attributes: processor status; key resources: system CPU usage, memory usage, page faults; run queues per SMP; scheduling priority of each process and each thread within a process; and current system configuration. The tuning parameter changes will take affect without requiring an operating system reboot. This single-point of control will require root access to make modifications, but only normal user privileges to monitor the system. Due to the large number of Purple system attributes and components, this single-point of control will be constructed to be fast and efficient when monitoring and modifying the entire Purple system. All system information and control functions will be presented in a hierarchical fashion.

2.2.4.3 Scalable Centralized Resource Data Repository (TR-2)

The Offeror will provide a scalable centralized resource data repository (CRDR), keeping track of the state of all system resources, their current usage policies, and a system error log. This facility and the system utilities/functions that depend on it, shall be constructed so that the centralized data repository does not become a single point of failure or contention (bottleneck) within the SMP cluster. In particular, updates to the CRDR and reading to and writing from the CRDR during system changes impacting at least 50% of the nodes (e.g., rebooting, major system

disruptions) will be done in parallel so as to not impede rapid system transitions. The degree of parallelism supported in the CRDR will be a system tunable parameter.

2.2.4.4 User Maintenance (TR-2)

The Offeror will provide a tool for managing user administration, including some means of integrating the namespace manager and the authentication server in order to facilitate adding, removing, and modifying users. In addition, the Offeror will provide a tool for managing groups, including initial creation of groups, modification of groups, and user membership in groups.

2.2.5 Parallelizing Compilers/Translators

2.2.5.1 Baseline Languages (TR-1)

The Offeror will provide fully supported implementations of Fortran95 (ISO/IEC 1539-1:1997(E), ISO/IEC 1539-2+:2000, ISO/IEC 1539-3:1998) see URL: http://www.nag.co.uk/sc22wg5/IS1539_family.html, C (ANSI/ISO/IEC 9899:1999; FIPS 160 or later), and C++ (ANSI/ISO/IEC 14882:1998, ISO/IEC 9945-1:1990/IEEE POSIX 1003.1-1990; ANSI/ISO-IEC 9899-1990 C standard, with support for Amendment 1:1994). Fortran95, C, and C++ shall be referred to as the baseline languages in this section. In addition, an assembler shall be provided. The Offeror shall provide the fully supported capability to build programs from a mixture of the baseline languages (i.e., inter-language subprocedure invocation must be supported).

2.2.5.2 Baseline Language 64b Pointer Default (TR-1)

The Offeror will provide compilers for the baseline languages that are configured with the default mode of producing 64b executables. A 64b executable will have all virtual memory pointers with 64b. All operating system calls will be available for use by 64b executables. All Offeror supplied libraries will provide 64b objects (versions of the API). Offeror's supplied software will be fully tested with 64b executables.

2.2.5.3 Baseline Language Standardization Tracking (TR-1)

The Offeror will provide a version of the baseline languages that is standard compliant within eighteen months after ANSI or ISO/IEC standardization, whichever occurs earlier. The Offeror is encouraged to adhere to the current proposed standard.

2.2.5.4 Common Preprocessor for Baseline Languages (TR-2)

The Offeror will provide the capability of preprocessing ANSI C preprocessor directives in programs written in any of the baseline languages.

2.2.5.5 Base Language Interprocedural Analysis (TR-2)

The Offeror will provide mechanisms to perform basic interprocedural analysis (e.g., variable cross-reference listing, COMMON block analysis, use/def analysis) for programs written in the baseline languages.

2.2.5.6 Baseline Language Compiler Generated Listings (TR-2)

The Offeror will provide baseline language compiler option(s) to produce source code listings that include information such as pseudo-assembly-language listings, optimizations performed and/or inhibitors to those optimizations on a line-by-line, code block-by-code block or loop-by-loop basis as appropriate, and variable types and memory layout.

2.2.5.7 C++ Functionality (TR-2)

The Offeror will provide an implementation of the ISO/IEC 14882 C++ standard compiler including: member function templates, partial specialization of classes, partial ordering of functions, name spaces including std::namespace for standard C++ libraries, and default template parameters. Standard C++ library including Standard Template Library and header files without ".h" extensions.

2.2.5.8 Cray Pointer Functionality (TR-2)

The Offeror will provide Cray style pointers implemented in an ANSI X3.9-1977 Fortran compliant compiler.

2.2.5.9 Baseline Language Support for OpenMP Parallelism (TR-1)

All the baseline languages (i.e., Fortran95, C and C++) compilers will support SMP parallelism through OpenMP Version 2.0, or later for Fortran95 and Version 1.0, or later for C/C++ directives or language constructs <http://www.openmp.org/specs/>. As an optimization feature, all the baseline language compilers will perform automatic parallelization. The baseline language compilers will produce symbol tables and any other information required by the debugger to enable debugging of OpenMP parallelized ASCI applications.

2.2.5.9.1 OpenMP Performance Interface (TR-3)

The baseline languages will implement portable OpenMP performance interface as specified by the OpenMP Forum within 12 month of ratification. The baseline languages will provide proper decoding and demangling of instructions and identifiers to support the mapping of results back to source code with respect to Fortran95 modules and C++ namespaces.

2.2.5.10 Debugging Optimized Applications (TR-2)

The baseline languages will produce symbol tables and any other information required by the debugger to enable the debugging, in the presence of "-O -g" code optimization, of ASCI applications. In particular, the baseline languages will provide a set of command line options that generate sufficient OpenMP optimized code and symbol table information so that the debugger can debug OpenMP threaded applications without loss of information about variables or source code context. See requirement 2.2.6.1.4.

2.2.6 Debugging and Tuning Tools

All debugging and tuning tools will be 64b executables and operate on 64b user applications by default.

2.2.6.1 Debugger for Cluster Wide Applications (TR-1)

The Offeror will provide an interactive debugger with an X11-based graphical user interface enabling single-point of control (multiple debugger invocations to control individual processes are not acceptable) that can debug cluster-wide applications with multiple parallel programming paradigms (e.g., message passing, OpenMP thread and process parallelism). In particular the debugger will be able to handle debugging jobs with one MPI task per CPU in the compute partition or one MPI task per node and one OpenMP thread per CPU on that node for every node in the compute partition. The parallel debugger will function at the initial source level (before any preprocessing) for programs developed with inter-mixed baseline languages. A command line interface will be available for sequential and parallel programs. The capability of

attaching/detaching the debugger to/from an executing (serial or parallel) program and modifying program state and continuing execution will be provided. If the code was not compiled for debugging, it is understood that access to source-level information will be limited. For MPI codes the debugger will display the status of message queues, such as number of pending messages and associated length, source, and sink at a breakpoint. For MPI codes the debugger will be able to breakpoint individual or groups or all tasks in a single GUI operation, step or continue an individual task, groups of tasks, or all tasks in a single GUI operation. For OpenMP threaded code the debugger will display the status of all threads, thread local and global variables, breakpoint individual or all OpenMP threads, step or continue individual or all OpenMP threads. Debugger functionality will include, but is not limited to: control of processes and threads (start/stop, breakpoints, and single-step into/over subprocedure invocations); examination of program state (stack tracebacks, contents of variables, array sections, aggregates, and blocks of memory, current states, registers, and source locations of processes); and modification of program state (changes to contents of variables, aggregates, and blocks of memory). The Etnus TotalView (http://www.etnus.com/Products/TotalView) debugger is highly preferred.

2.2.6.1.1 Visual Representation of Data (TR-2)

The Offeror will provide a parallel debugger capable of displaying multiple visual representations of values in a matrix or 2-D array section (e.g., bitmap showing elements exceeding a threshold value, colormap, surface map, contour map) with zoom and pan capability for the visual displays to facilitate scaling and display of large arrays. This functionality will be provided for all baseline languages. The Offeror will provide a conditional data filtering capability for large data sets integrated with data display functions, both textual and graphic. All visual capability will be invoked directly from the debugger.

2.2.6.1.2 Fast Conditional Data Watchpoints and Breakpoints (TR-2)

The Offeror provided debugger will support fast conditional data watchpoints and breakpoints in all the baseline languages. That is, an implementation for memory or source code breakpoints which does not degrade the application runtime within the debugger more than a factor of 1.5 for conditional watch points or conditional breakpoints relative to the runtime without conditional data watchpoints or conditional source code breakpoints set.

2.2.6.1.3 Memory Leak Debugging (TR-2)

The Offeror will provide the capability of reporting memory access errors and pointing to the offending source line in the baseline languages. Memory access errors reported will include: accessing/freeing beyond allocated block; accessing/freeing unallocated blocks; memory leaks (accumulated memory chunks from malloc calls that can no longer be accessed or freed); and uninitialized memory read/write.

2.2.6.1.4 Debugging Optimized Applications (TR-2)

The parallel debugger will, in the presence of "-O -g" code optimization, provide a fully supported mechanism for reporting information on program state (stack traceback, access to variables that have not been eliminated), breakpoints at basic block boundaries, single-stepping at the basic block level, and stepping over subroutines. In particular, the debugger will be able to debug OpenMP threaded applications without loss of information about variables or source code context.

2.2.6.1.5 Debugger Expression Evaluator (TR-2)

The parallel debugger shall have an evaluator capable of calculating the results of simple expressions (in "free floating" C and/or Fortran95) such as values of conditionals, indirect array references, etc. It is also desired that the evaluator handle the supported languages. This might be a language interpreter, but for the purposes of user code to be executed at breakpoints, or watchpoints, some form of compiled code is more desirable to make impact on execution smaller.

2.2.6.1.6 Parallel Debugger Barrier-Points (TR-2)

The parallel debugger will have an expanded breakpoint functionality for control of parallel processes by setting a "barrier-point." With a barrier point, the process will be held until all processes reach the same point, not responding to "start" commands until the barrier point is satisfied, or released.

2.2.6.1.7 Post-Mortem Debugging (TR-2)

The debugger will have a fully supported implementation of some mechanism for invoking the debugger for examining the final state of a program that failed ("postmortem debugging"). Facilities for modifying program state and/or continuing execution need not be available in this mode. If the code was not compiled for debugging, it is understood that access to source-level information will be limited.

2.2.6.1.8 Symbol Table (TR-2)

The time to initialize the debugger on an application with a 50 MB symbol table will be less than a minute longer than the time to initialize the debugger on the same number of processors, but with no symbol table.

2.2.6.1.9 Data Aggregation (TR-2)

The parallel debugger will have a capability of accumulating the local values of variables that are replicated across multiple threads/processes, and presenting a condensed summary within a single window. In addition, where distributed arrays are supported by the programming model, the debugger will have the capability of gathering the elements of a distributed 2-D array and presenting them in a single table/visualization.

2.2.6.2 Stack Traceback (TR-2)

The Offeror will provide runtime support for stack traceback error reporting. Critical information will be generated to stderr upon interruption of a process or thread involving any trap for which the user program has not defined a handler. The information will include a source-level stack traceback (indicating the approximate location of the process or thread in terms of source routine and line number) and an indication of the interrupt type.

2.2.6.3 Lightweight Corefile API (TR-2)

The Offeror will provide the standard lightweight corefile API, defined by the Parallel Tools Consortium, to trigger generation of aggregate traceback data like that described in 2.2.6.2. The specific format and command-line and graphical browsers for the lightweight corefile facility are defined by the Parallel Tools Consortium.

2.2.6.4 Profiling Tools for Cluster Applications (TR-1)

The Offeror will provide tools for profiling CPU time distribution from all processes or threads in a parallel program, at the levels of subprocedures and coarse blocks (e.g., large loops). The tools will include a capability for restricting the amount of profiling data collected to certain portions of the source code (e.g., a specific subset of procedures), through the use of compiler directives, API or command-line switches. The tools will display the profiling data in a GUI showing the CPU time distribution on a source code level. The granularity of this display will be down to the source code block level. The statistics gathering and GUI functions will be usable when profiling an MPI/OpenMP threaded application running over an entire Purple system.

2.2.6.5 Event Tracing Tools for Cluster Applications (TR-1)

The Offeror will provide event tracing tools for cluster wide applications. Distributed mechanisms for generating event records from all process and threads in the parallel program will include timestamp and event type designators and will be formatted in a self-defining data format. This functionality must be provided for all baseline languages. The event tracing tool API will dynamically activate and deactivate event monitoring during execution from both within and outside a process. By default, event tracing tools will not require dynamic activation to enable tracing.

2.2.6.5.1 Binary Event Trace Ouput Translation (TR-1)

Offeror will provide a supported and documented utility that converters binary event trace files to human readable ASCII text files. ASCII output format will allow for easy "grep-ing" or "gawk-ing" out of individual or groups of events. Offeror provided documentation will include an explanation of every event type and all their encoded fields.

2.2.6.5.2 Message-Passing Event Tracing (TR-2)

The Offeror will provide a fully supported implementation of some mechanism for tracing message sends, receives, and synchronizations, including non-blocking messages, for the MPI libraries.

2.2.6.5.3 Lightweight Message-Passing Profiling (TR-2)

The Offeror will provide a lightweight, scalable profiling library for MPI that captures only timing statistics about each MPI task. Instead of capturing entire traces, this tool captures limited data that includes min/max/cumulative time and a call count for each MPI callsite on a per task basis.

2.2.6.6 Performance Statistics Tools for Cluster Applications (TR-1)

The Offeror will provide performance statistics tools, whereby performance measures obtained for individual threads or processes are reported and summarized for the cluster wide application. There will be a mechanism for capturing the statistics and storing them for later analysis/viewing. In addition to the hardware performance monitors (section 2.1.2.8), the Offeror will support: lock or critical section usage; page faults; I/O and communications (in terms of bytes sent/received). The measures will also include a summary of memory usage (actual allocations, not memory references) and a mechanism to relate that back to baseline language source code.

2.2.6.7 Cluster Wide Application Development Tool API (TR-2)

The Offeror supplied APIs in sections 2.2.6.5, 2.2.6.5.1 and 2.2.6.5.3 will provide a means for dynamically activating and deactivating, reading and resetting data for profiling, trace, and

performance statistic instrumentation. The API will dynamically control, activate and deactivate the instrumentation of portions of an application (from specific code lines to the entire application) during execution based on dynamic conditions within the running process. The University's strong preference is for the Open Source DPCL library and runtime infrastructure (http://oss.software.ibm.com/developerworks/opensource/dpcl/).

2.2.6.8 Cluster Wide Application Development Tool GUI (TR-1)

The Offeror shall provide a scalable GUI tool or set of GUI tools that display MPI trace data (as defined in 2.2.6.5 and 2.2.6.7) generated from MPI/OpenMP threaded applications. The statistics gathering and GUI functions for MPI tracing (section 2.2.6.5) shall be usable when applied to an MPI/threaded application running over an entire Purple system.

2.2.6.9 Timer API (TR-2)

The Offeror will provide an implementation of the Parallel Tools Consortium API for interval wall clock and for interval CPU timers local to a thread/process. The interval wall clock timer mean overhead shall be less than 250 nanoseconds to invoke and shall have a resolution of less than 1 processor clock period. The system and user timers mean overhead shall be less than 1.5 microsecond to invoke and shall have a global resolution of 10 milliseconds (i.e., this wall clock is a system wide clock and is accurate across the system to 10 milliseconds).

2.2.7 Applications Building

2.2.7.1 Linker and Library Building Utility (TR-1)

The Offeror will provide an application linker with the capability to link object and library modules into a dynamic and static executable binary. By static execution binary we mean a binary that has all user object modules and libraries statically linked when the binary is created. By dynamic executable binary we mean that all the user object modules and static libraries are linked at binary creation, but that the user and system dynamic libraries are loaded at runtime on a demand basis. The linker and library building utility will produce executable binaries and static and dynamic load libraries that are 64b by default. In addition the linker will be capable of re-linking selected portions of an application (i.e., replace specific objects within the binary) rather rebuilding the executable binary from scratch. The Offeror will include a facility to build and incrementally update static and dynamic libraries of object modules. The loader will be able to generate a full link listing of the load indicating at a minimum: which object file and original source file every function was taken from; which system functions were loaded from what library; complete memory map including function start points and the layout of all static and dynamic variables. If the microprocessor architecture possesses a memory reference model that includes segments, then the memory layout shall be delineated by segment. The linker will provide the user with the capability of managing the memory layout by specifying the order in which libraries are loaded, the order variables and functions are loaded, etc. The compiler/linker combination will provide users the ability to control the placement of underscores (_), or other Offeror provided name mangling mechanisms, in front of or behind of externally visible variable and function names.

2.2.7.2 Make Utility (TR-1)

The Offeror will provide a make utility.

2.2.7.3 Parallel Make (TR-2)

The Offeror will provide a UNIX make facility that can utilize parallelism in performing the tasks in a makefile.

2.2.7.4 Source Code Management (TR-2)

The Offeror will provide a set of tools for the management of source code in a multiple programmer project environment (e.g., SCCS, USM, RCS, CVS).

2.2.7.5 Dynamic Processor Allocation (TR-2)

By setting various UNIX shell environment variables and/or interactive or batch command line options, users shall be able to run threaded applications compiled from any combination of the baseline languages exploiting automatic parallelization, SMP compiler options, and/or MPI parallel application on varying numbers of processors and/or nodes without recompilation or relinking.

2.2.8 Application Programming Interfaces

All Offeror supplied APIs will support 64b executables and be fully tested in 64b mode. In particular, sPPM and UMT2000 will be 64b executables that utilize both OpenMP and MPI styles of parallelism in a single 64b executable and run successfully with at least 6 GiB of user memory per user process over the entire machine.

2.2.8.1 Optimized Message-Passing Interface (MPI) Library (TR-1)

The Offeror shall provide a fully supported implementation of the MPI-2 standard, as defined by the most recent MPI-2 specification of the MPI forum. The Purple system shall be delivered with an optimized MPI-2 version compliant with current MPI-2 standard as defined by:
http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html
The MPI library will be highly optimized in the sense that it will effectively and efficiently utilize all available hardware on the SMP cluster. In particular, the MPI library will operate transparently and directly on the Purple cluster interconnect network (i.e. not over TCP/IP or some other intermediate software layer). If the cluster interconnect network has multiple planes, then the MPI library will utilize the multiple planes to increase effective single task and node aggregate MPI off-node performance. The MPI library will be architected and implemented to minimize latency for small messages and maximize bandwidth for large messages under normal operating conditions. The MPI library shall be architected and implemented to utilize shared memory for communications between MPI tasks on a single node and single SMP (both - this is important). The MPI global operations such as MPI_Barrier, MPI_Allreduce, MPI_Reduce, MPI_Broadcast will be architected and implemented to utilize hardware reduce and broadcast features of the cluster interconnect and take advantage of shared memory on a node and SMP (both) to do Barriers, reductions and broadcasts first between task on a node and then between nodes as separate steps. It is insufficient to utilize shared memory solely for fast task to task communications in these operations. The MPI library will support up to one MPI task per CPU in the entire Purple system. The MPI buffers will be managed so that an application can set the amount of buffer space required for point-to-point and all-to-all communications. In particular, if an application guarantees that receives are posted before sends, then it will be possible to avoid MPI buffers completely. The Offeror shall provide (electronic) written documentation that describes the performance features of the MPI implementation for each software release on the proposed Purple hardware. All environmental settings that impact MPI operation, buffering and

performance and their impact to 64b user applications performance shall be tested and their effectiveness and reliability documented.

2.2.8.1.1 MPI Profiling Interface (TR-2)

The MPI library shall provide an MPI profiling interface.

2.2.8.2 Low Latency Remote Memory Access API (TR-2)

The Offeror shall provide a fully supported (API) that allows user applications to directly program the hardware remote memory access hardware (see section 2.1.1.9). This interface will be low latency, support a large number of outstanding operations for each CPU in the node, be thread safe and allow a single 64b executable user application to remote get or put any virtual memory location in the user application on the cluster within the limits of the physical address space (see section 2.1.3.4).

2.2.8.3 Visual Display of Message Traffic (TR-2)

The Offeror will ensure the tools described in sections 2.2.6.5.1 and 2.2.6.8 display message traffic and program state in a time-space diagram (like that generated by Pallas Vampir or ANL Jumpshot) or other mutually agreeable presentation methodology. See URL: http://www.pallas.com/pages/vampir.htm and http://www-unix.mcs.anl.gov/perfvis/software/viewers/jumpshot-3/index.html.

2.2.8.4 Parallel I/O API (TR-2)

The Offeror will provide a fully supported implementation of the POSIX file I/O (1:N parallel) API supporting concurrent file I/O via CWFS. CWFS will support the POSIX "atime, ctime, mtime" functions, but the implementation will not update the "atime, ctime, mtime" immediately as required by the standard. All threads and processes will have access to a common file system. Files and directories in this system will be available under the same names to each process or thread. The file system will support concurrent access by multiple threads or processes to a single logical file. When such accesses by different threads or processes are to disjoint regions of the file, the file system will support efficient parallel access when the physical layout of the file on storage resources permits it. When such accesses by different threads are to overlapping regions of a single logical file, the file system will support an automatic locking mechanism (not requiring special action by the user) that ensures such overlapping accesses exhibit sequential consistency.

2.2.8.4.1 MPI I/O (M:N) Parallel I/O API

The Offeror will also provide a complete implementation of the parallel file I/O interface defined in the MPI-2 ("MPI-IO") standard, as released in July 1997 (or any later revision, http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html). This interface will access files in the file system described in the previous paragraph. Individual files created and manipulated through the MPI-IO interface will be directly accessible as single logical files through the POSIX file interface. Individual files created and manipulated through the POSIX file interface will be directly accessible as single logical files through the MPI-IO interface.

2.2.8.4.2 Parallel I/O Buffer Attributes

The supplied CWFS I/O API's will have user configurable buffer lengths. CWFS will automatically flush all user buffers associated with a job upon normal completion or explicit

call to "abort()" termination of the job. CWFS will also have an API for application invoked flushing of all user buffers.

2.2.8.4.3 Parallel I/O Sequential Write

Output from a Purple System wide interactive or batch parallel job directed to STDOUT/STDERR will be atomic on a line by line basis and will be prefixed with a label indicating which MPI task performed the write.

2.2.8.4.4 Asynchronous Parallel I/O

The supplied CWFS and MPI I/O parallel I/O products will support asynchronous read/write operations.

2.2.8.5 Graphical User Interface API (TR-1)

The Offeror will provide the standard X11R6 and Motif 1.2, or current versions, applications, servers and API libraries.

2.2.8.6 Visualization API (TR-2)

The Offeror will provide OpenGL 1.3, or current version, (http://www.opengl.org).

2.2.8.7 Math Libraries (TR-2)

The Offeror will supply highly optimized mathematical libraries, callable from all baseline languages, for serial (single processor) node or SMP and cluster wide applications.

2.2.9 Operating System Security (TR-2)

The Offeror will provide security functionality where access to the system will be controlled by identifying and authorizing the user or by checking the validity of forwarded credentials. All users will be authenticated before access is permitted. Successive logon attempts will be controlled by denying access after multiple (maximum of 5) unsuccessful logon attempts by the same user.

2.2.9.1 Login Information (TR-2)

Users will be notified upon successful login of the following information: date and time of last successful login; and where the operating system provides the capability, number of unsuccessful attempts.

2.2.9.2 Audit Capability (TR-1)

A record of each user login and logoff will be maintained. In addition, the following information will be maintained as an audit record: use of authentication changing procedures; unsuccessful logon attempts; and blocking of a user, and the reason for the blocking.

2.2.10 Compliance with DOE Security Mandates (TR-1)

DOE Security Orders have changed over time. From time to time, these mandates cause the University and/or it's Offerors, to fix bugs or implement security features in vendor operating systems and utilities.

The Offeror will handle bug fixes as follows: A "bug" is interpreted to mean that a product does not perform as documented. If a "bug" is discovered, there are standard reporting procedures which must be followed for tracking purposes. Additionally, the report from the University would be escalated by the Offeror or the ASCI program office to achieve priority resolution.

For implementation of security features in Offeror's operating system and utilities, Offeror requires written notification of the changes to DOE Security Orders or their interpretation that would force changes in system functionality. If the request for change would result in a modification consistent with standard commercial offerings and product plans, the Offeror will perform the change. If the change is outside the range of standard offerings, the Offeror will make the operating system source code available to the University (at no additional cost, assuming the University holds the proper USL and other prerequisite licenses) under the terms and conditions of the Offeror's standard source code offering.

2.2.11 Cluster Environment On-Line Document (TR-2)

The Offeror shall supply hardcopy and on-line documentation by http based mechanism for all major hardware and software subsystems viewable from standard Netscape or Explorer Web browsers.

2.2.12 Early Access to Purple Software Technology (TR-1)

The Offeror may propose mechanisms to provide the University early access to Purple software technology and to test software releases and patches before installation on Purple that includes other steps before installing the software on EDTV and the dual boot environment on Purple.

End of Section 2.0

3.0 Purple C Option (MO)

The ASCI applications efforts have demonstrated the capability to perform 3D simulation of both primary and secondary aspects of nuclear weapons functioning. As a result of these challenging successes, the ASCI applications have been rapidly adopted by the Stockpile Stewardship Program (SSP) for Directed Stockpile Work (DSW) and various weapon Life Extension Programs (LEPs). This trend of increased use of ASCI applications for day-to-day use in the DSW and LEPs indicate that much more capability is required by ASCI in the late 2004 through 2007 timeframe. In addition, it is anticipated that the ASCI applications milestones will increase in complexity and runtime in the 2005-2007 timeframe, requiring additional computational capability. Thus, the ASCI program requires an option to scale the Purple SMP cluster up to peak performance of 100 teraFLOP/s and peak plus sustained performance 130 teraFLOP/s (with reasonable effort to achieve 145 teraFLOP/s) based on the performance of the sPPM and UMT2000 marquee demonstration codes in this timeframe. This option, if executed, will take effect no later than the time of contract award.

The specific hardware and software requirements of the Purple C mandatory optional system are delineated in section 3.1, Purple C Hardware Requirements, and section 3.2, Purple C Software Requirements. Rather than replicate all of the Purple requirements in this section, we include all of the Purple requirements and their associated priorities (TR-1, TR-2 and TR-3) and then note differences for the Purple C mandatory option. This fully functional Purple C system must be useful in the sense of being able to deliver a large fraction of peak performance to a diverse scientific and engineering workload. It must also be useful in the sense that the code development and production environments are robust and facilitate the dynamic workload requirements.

In addition to the hardware and software requirements, the Offeror will deliver any additional features consistent with the objectives of this project and Offeror's Full-Term Plan of Record, which the Offeror believes will be of benefit to the project.

Development of the Purple C mandatory option, should this option be executed at the time of contract award, shall comply with the requirements identified in section 8.0, Project Management.

3.1 Purple C Hardware Requirements

All of the Purple Hardware requirements (section 2.1) apply to the Purple C system. The following requirements supercede section 2.1.

3.1.1 Scalable Cluster of SMPs

3.1.1.1 Purple Scalable SMP Cluster (MR)

The Offeror shall provide a Purple teraFLOP/s system composed of multiple Shared memory Multi-Processors (SMPs) connected via a scalable intra-cluster communications technology. The system shall have a peak performance of at least one hundred teraFLOP/s (1.0×10^{14} floating point operations per second) and a peak plus sustained performance of at least one hundred thirty teraFLOP/s (1.3×10^{14} floating point operations per second) on the two SSP marquee benchmarks, sPPM and UMT2000. This is nominally 130% of the target peak.

3.1.1.2 Purple Component Scaling (TR-1)

In order to provide the maximum flexibility to the Offeror in meeting the goals of the ASCI project the exact configuration of the Purple SMP scalable cluster is not specified. Rather the

Purple configuration is given in terms of lower bounds on component attributes relative to the peak performance of the proposed configuration. The Purple SMP scalable cluster configuration shall meet or exceed the following parameters:

- Memory Size (Byte/FLOP/s) ≥ 0.5
- Globally Addressable User Disk Space (Byte/FLOP/s) ≥ 20
- Memory Bandwidth (Byte/s/FLOP/s) ≥ 1
- Intra-Cluster Network Aggregate Link Bandwidth (Bytes/s/FLOP/s) ≥ 0.1
- Intra-Cluster Network Bi-Section Bandwidth (Bytes/s/FLOP/s) ≥ 0.05
- System Sustained Productive Disk I/O Bandwidth (Byte/s/FLOP/s) ≥ 0.001
- Cluster High Speed External Network Interfaces (bit/s/FLOP/s) ≥ 0.00125

Example: For a 100 teraFLOP/s peak system, requirement 3.1.1.2 specifies that the system shall have at least 50 TiB of memory, 2.0 PB globally addressable user disk, 100 TB/s memory bandwidth, 10.0 TB/s intra-cluster network aggregate link bandwidth, 5.0 TB/s intra-cluster networking bi-sectional bandwidth, 100 GB/s system sustained productive disk I/O bandwidth and 125 Gb/s external networking.

3.1.1.3 Additional Applications Memory for Cluster (MO)

As a separately priced option, the Offeror will install 25.0 TiB ($25x2^{40}$ Bytes) of additional memory in the Purple 100T system utilizing the same density and performance memory components as configured in the base system. This option shall include sufficient additional disk to meet the swap space requirement defined in Requirement 2.1.4.4. This requirement supercedes Requirement 2.1.1.10.

3.1.1.4 Additional Applications Memory for Compute SMP (MO)

As a separately priced option, the Offeror will install sufficient memory to double the memory in one Purple 100T compute SMP utilizing the same density and performance memory components as configured in the base system. This option shall include sufficient additional disk to meet the swap space requirement defined in Requirement 2.1.4.4. This requirement supercedes Requirement 2.1.1.11.

3.2 Purple C Software Requirements

All of the Purple Software requirements (section 2.1.5) apply to the Purple C system. The following requirements supercede section 2.1.5.

3.2.1 Checkpoint/Restart Minimum I/O Rate (TR-2)

A parallel job utilizing 25.0 TB of system memory will be checkpointed/restarted to/from local disk or CWFS in less than 500 seconds (50.0 GB/s delivered overall). This requirement supercedes Requirement 2.2.1.8.1.

End of Section 3.0

4.0 EDTV High-Level Technical Requirements

The ASCI program requires early access to stable code development platforms for the rapid development of Stockpile Stewardship applications for the Purple system. It is imperative that the Early Deployment Technology Vehicle (EDTV) code development environment present the same hardware and software model to code developers that will be available on the Purple. In particular, hardware issues such as memory hierarchy and message passing support must present the same issues to application performance that will be experienced in the Purple. As an example, if high cache utilization is required for application performance on Purple then the EDTV must present the same performance challenges and opportunities for optimization.

It is envisioned that the Early Deployment of Technology Vehicle (EDTV) will come from the subcontractor's current (or very near term) product offering and provide, to the maximum extent possible, a code development environment congruent with that of Purple.

In specifying the EDTV system the University was motivated by four factors:

1. provide a delivery vehicle that focuses the efforts of the ASCI/Offeror partnership before the delivery of Purple;
2. provide an early hardware and software code development environment that closely resembles the Purple system;
3. provide additional capability beyond what is currently available at LLNL;
4. provide a vehicle for early testing of Purple hardware and/or software.

This EDTV system will be delivered to LLNL with an optional second delivery as directed by the University.

Due to the classified and unclassified ASCI programmatic requirements it would be ideal if this EDTV would function in both the classified (RED) and unclassified (BLACK) networking partitions. The best way to accomplish this is to have the EDTV system statically split between the RED and BLACK computing partitions. The two portions of EDTV will be combined into a single unclassified machine after Purple is accepted. The Offeror can propose a split within limits: at least 20% and at most 30% of the EDTV system should be in the BLACK environment. This will provide the partnership with the ability to test the hardware, software, networking and unclassified ASCI applications in an unclassified environment. This unclassified portion will allow the partnership to work out issues and test software patches more effectively and quickly.

The specific hardware and software requirements of the EDTV system are delineated in section 3.1, EDTV Hardware Requirements, and section 3.2, EDTV Software Requirements. Rather than replicate all of the Purple requirements in this section, we include all of the Purple requirements and their associated priorities (TR-1, TR-2 and TR-3) and then note differences for EDTV. However, during the course of executing the Purple contract it is anticipated by the University that more functionality will be provide over time. The proposal preparation instructions indicates where in the RFP response the Offeror shall delineate in detail the proposed schedule of hardware and software deliverables and in particulars the EDTV and technology refresh deliverables.

In addition to the hardware and software requirements, the Offeror will deliver any additional features consistent with the objectives of this project and Offeror's Full-Term Plan of Record, which the Offeror believes will be of benefit to the project.

4.1 EDTV Hardware Requirements

All of the Purple Hardware requirements (section 2.1) apply to the EDTV system(s). The following requirements supercede section 2.1.

4.1.1 EDTV-5 SMP Cluster (MO)

The Offeror shall propose a fully configured, complete and functional EDTV SMP cluster with at least 5.0 teraFLOP/s peak performance.

4.1.2 EDTV-20 SMP Cluster (MO)

The Offeror shall propose a fully configured, complete and functional EDTV SMP cluster with at least 20.0 teraFLOP/s peak performance.

4.1.3 EDTV RED/BLACK Static Split (TR-1)

The Offeror will propose a RED/BLACK static split of the EDTV-5 (and EDTV-20) configurations that has at least 20% of the resources in the BLACK environment and at most 30% of the resources on the BLACK environment. The remainder of the EDTV-5 (and EDTV-20) configuration will be in the RED environment. There must be no shared resources between the RED and BLACK portions of the EDTV-5 (and EDTV-20) clusters.

4.1.4 Node Shared Main User Memory Size (TR-1)

The node shared main user memory available to applications shall be larger than 14 GB. User memory available to applications shall be measured by a single MPI application with one MPI task on every CPU in the compute partition performing all to all communications as the amount of user data and code memory space available to the application. This specifically excludes memory required by the Operating System and buffers, required daemons (e.g., CWFS client, CWARM scheduler) and other system services and MPI buffers.

4.2 EDTV Software Requirements

All of the Purple Software requirements (section 2.1.5) apply to the EDTV system(s). The following requirements supercede section 2.1.5.

NONE

End of Section 4.0

5.0 Visualization Requirements (MO)

Visualization plays a critical role in the evaluation and interpretation of simulation results. The ASCI program requires that the data being generated by the EDTV and Purple systems be accessible for visualization purposes. The ASCI program requires the option of tight integration of ASCI EDTV, Technology Refresh (if proposed), and Purple systems with visualization solutions. This is due to the fact that it may be impracticable or impossible to duplicate datasets of the magnitude of those generated on EDTV or Purple systems on other systems. One way to facilitate this integration is to add additional SMPs on the EDTV and Purple systems themselves, dedicated to the support of visualization application. This allows those dedicated visualization SMPs direct access to the data residing on the EDTV and Purple CWFS.

A simplified visualization data pipeline would include the reading of simulation data from the disks attached to EDTV and Purple, the computation/extraction of graphics primitives from that data, and the rendering/display of such primitives to users' offices or to shared collaborative facilities. In all cases, the final display location would be located at some distance from the host platform (potentially many miles) and in some cases, the display may take the form high high-resolution/tiled displays which would have aggregate pixel counts many times higher than a single desktop display. Both types of destinations should be supported from the same physical hardware. We require simultaneous access by at least five desktop users (2 Mpixel each) and a PowerWall with at least 20 Mpixel (utilizing 1.3 Mpixel tiles). Thus, the aggregate number of display pixels that shall be simultaneously driven from the EDTV and Purple visualization SMPs is at least 30 Mpixel. It may also be possible to apply the aggregate rendering capacity of the system to a single desktop display, improving the rendering capacity for that single display. The system will support any mix of aggregated capacity and increased image size options, switching between the various output configuration options on a job by job basis.

5.1 Visualization Hardware Requirements (TR-1)

The integration of visualization capability directly into EDTV and Purple has some common hardware requirements: sections 5.1.1 through 5.1.4. There are two basic scenarios for visualization nodes integrated into the EDTV and Purple platforms, SMPs with (section 5.1.5) or without (section 5.1.6) hardware graphics accelerators. These scenarios differ in their basic hardware requirements. **The Offeror must bid at least one, and preferably both, of the two options to be compliant with the section 5.0 mandatory option requirement.** In any case, note that there is no requirement that the SMPs used for visualization be of the same type or configuration as the SMPs utilized in the rest of the platform. From studying the use of visualization in achieving ASCI programmatic objectives, we anticipate that a visualization system with 10% of the processing power (teraFLOP/s) of the EDTV or Purple systems is required, in addition to the EDTV or Purple systems.

Example: A 20 teraFLOP/s EDTV system requires an additional 2 teraFLOP/s peak computational capacity in the EDTV cluster for visualization. A 60 teraFLOP/s Purple system requires an additional 6 teraFLOP/s of computational capacity in the Purple cluster for visualization.

5.1.1 Visualization Hardware Cluster Interconnect (TR-1)

The EDTV or Purple SMPs will be connected to the EDTV or Purple, respectively, cluster interconnect. That is, the visualization SMPs are in addition to the EDTV or Purple core SMPs. Adding SMPs to the EDTV or Purple systems for visualization implies that the EDTV or Purple interconnects have expansion capability to accommodate the additional SMPs.

5.1.2 Direct Access to CWFS (TR-1)

A fundamental motivation for proposing a visualization system tightly coupled to the EDTV and Purple systems is to avoid the need to duplicate and transfer users' data off the simulation systems. Thus, the added visualization SMPs will have the same methods of access to the EDTV and Purple CWFS as the other SMPs in the system. Delivered read bandwidth to ASCI visualization applications through the cluster interconnect to the CWFS will be at least 50.0 MB/s/CPU. Note that this I/O scaling is different from the core SMP I/O scaling requirement.

5.1.3 Visualization Off-SMP External Networking Bandwidth (TR-2)

The fact that the users' desktops will be located at some distance from the EDTV and Purple systems effect the requirement for a mechanism for bringing data off of the SMPs generating the visualization data. This transfer may take the form of image streams or it may take the form of streams of polygonal geometry in the case of interactive visualization. Capacity for moving entire datasets directly from the visualization SMPs over the external TCP/IP network to external systems must be provided on EDTV and Purple. The delivered TCP/IP bandwidth required for interactive use (using an image transfer model) is at least 0.3 Gb/s/1 Mpixel display screen. To meet the requirement of driving an aggregate of at least 30 Mpixels of display (section 5.0), the simultaneous delivered off-visualization system external network TCP/IP bandwidth required is at least 9.0 Gb/s. This bandwidth will be in addition to the core external network bandwidth of the EDTV and Purple (section 2.1.1.2) systems. Note that this directly attached external networking is a different scaling of the external networking bandwidth requirement than for the rest of the EDTV and Purple systems.

5.1.4 Hardware Rendering SMP Memory Size (TR-1)

The visualization SMPs will have at least 1.0 GiB of memory per CPU in the SMP. Note that this is a different memory scaling requirement than the core SMPs of EDTV and Purple.

5.1.5 SMPs Augmented with Hardware Graphics Accelerators (TR-1)

In this scenario, the visualization SMPs include custom graphics rendering hardware. This is the preferred scenario. In this case, the SMPs perform I/O operations, primitive extraction/routing and rendering. The image pixels will be read from the frame-buffers and transmitted to remote clients by DVI extension. An additional, but lower desirability (TR-3) mechanism will be to route the video output directly to a University supplied analog video switch, which will in turn route to office displays

5.1.5.1 Graphics Rendering Hardware (TR-2)

The minimum acceptable graphics hardware accelerator includes: a 24 bit double buffered frame buffer capable of at least 2048x1536 pixel resolution supporting at least a 24bit depth buffer, OpenGL 1.3 or later support for rendering in hardware (including 3D texture mapping, transform and lighting), and a rendering rate of at least 30 Mpolygon/s. One or more such cards may located in each node subject to bandwidth constraints (section 5.1.5.3). In addition to supplying the graphics accelerator hardware, a hardware or software mechanism will be provided to allow for the aggregation of the multiple graphics accelerator hardware outputs, both for use with large pixel count displays and for scaling the rendering capacity of a single output.

5.1.5.2 Hardware Rendering SMP CPU Scaling (TR-2)

At least two (2) CPUs within the SMPs are required for each graphics hardware accelerator. If the graphics hardware accelerator (section 5.1.5.1) does not support transform and lighting in hardware, then four (4) CPUs within the SMPs are required for each graphics accelerator.

5.1.5.3 Hardware Rendering SMP Cluster Interconnect Bandwidth (TR-2)

SMP cluster interconnect bandwidth requirement is driven by the need to stream geometry directly to the graphics accelerators. Sufficient bandwidth into the SMP shall be provided to keep the graphics hardware busy, particularly when using API infrastructures like Chromium (http://chromium.sourceforge.net). The proposed SMP cluster interconnect will deliver to ASCI visualization applications utilizing the Chromium library with TCP I/P transport or any other Chromium supported transport layer at 500 MB/s of bandwidth for graphics primitives over the cluster interconnect to each SMP per graphics card in the SMP. Note that this is a different cluster interconnect scaling than that required for the core SMPs.

5.1.6 Visualization SMPs Without Hardware Graphics Accelerators (TR-3)

In this scenario, the visualization SMPs do not include custom graphics rendering hardware. In this case, the SMPs will generally be used for high performance I/O operations and graphical primitive extraction. The resulting primitives would be transferred off the system to remote rendering facilities. In some cases, the application codes might also perform parallel rendering in software on the nodes. The requirements are as follows:

5.1.6.1 Bandwidth Requirements (TR-1)

Aggregate delivered TCP/IP bandwidth off each visualization SMP to the Tri-Laboratory WAN will be at least 80 MB/s/CPU (based on data rates to send composite capable imagery tiles at interactive frame rates). This bandwidth will be in addition to the core external network bandwidth of the EDTV and Purple (section 2.1.1.2) systems. Note that this is a different scaling of the external networking bandwidth requirement than for the rest of the EDTV and Purple systems.

5.2 Visualization Software Requirements

The nature of visualization applications makes unique system and applications software demands as well.

5.2.1 "Interactive" Job Queues (TR-2)

The visualization SMPs will be on the same cluster interconnect as the core SMPs of the EDTV or Purple cluster interconnect, but will be managed in a different fashion from the rest of the system. Interactive usage patterns require more immediate allocation of nodes. A user will be sitting at their desktop waiting for the visualization process to be invoked (interactive usage). Minimizing that wait time is crucial. Highest priority work on the visualization SMPs will be interactive visualization jobs. In addition to supporting interactive jobs, the visualization SMPs will be used to run lower priority batch (visualization and computation) jobs. Batch jobs will be submitted to the visualization SMPs via CWARM and will be managed by the CWARM on the visualization SMPs. Interactive visualization jobs will run with higher CWARM priority and will automatically preempt any existing batch jobs running on the visualization SMPs. This facility will allow the visualization SMPs to be utilized for lower priority batch computation while preserving their interactive role. The resource

management system will be tunable so that interactive job launch and queuing mechanism controlling the visualization nodes can be tuned towards providing interactive users immediate access to nodes.

5.2.2 High Performance Visualization I/O Access Patterns (TR-2)

Visualization applications do not commonly share the same access patterns as simulations. Since visualization is often performed on a smaller number of SMPs than the original simulation, datasets may not fit in the memory space of the visualization process. Looking at the progression of data over time is also very key to visualization, so the applications will be looking to access data across timesteps as well. As a result, visualization I/O patterns tend to be read heavy and tend to access multiple smaller pieces of the data (e.g. one or more physical variables such as temperature and/or pressure) within the many files comprising a single data set. The CWFS will deliver the specified read rate (section 5.1.2) for random access patterns of 2MB or larger contiguous blocks of file data to ASCI visualization applications running on the visualization SMPs.

5.2.3 Graphics API Support (TR-1)

In order to perform rendering operations on the SMPs, OpenGL 1.3 (http://www.opengl.org) or later will be supported.

End of Section 5.0

6.0 Integrated System Features

The following requirements deal with the functional aspects of the integrated EDTV, technology refresh and Purple systems.

6.1 Reliability, Availability, Serviceability (RAS) and Maintenance

6.1.1 Capability Application Reliability (TR-1)

A user application job spanning 80% of the SMPs in the cluster (including compute, login, service, I/O and visualization SMPs) will complete a run with correct results that utilizes 200 hrs of system plus user CPU time per CPU in at most 240 wall clock hours without human intervention. A user application job spanning 30% of the SMPs in the cluster (including compute, login, service, I/O and visualization SMPs) will complete a run with correct results that utilizes 200 hrs of system plus user CPU time per CPU in at most 220 wall clock hours without human intervention.

6.1.2 Power Cycling (TR-3)

The system will be able to tolerate power cycling at least once per week over its life cycle.

6.1.3 Hot Swap Capability (TR-2)

Hot swapping of failed Field Replaceable Units (FRUs) will be possible without power cycling the cabinet in which the FRU is located The service strategy will ensure that a granular FRU structure is implemented. A granular FRU structure means that the maximum number of components (such as CPUs, memory, disks and power supplies) contained in or on one FRU will be less than 0.1% of the components of that type in the system for system components with at least 1,000 replications.

6.1.4 Production Level System Stability (TR-2)

The system (both hardware and software) will execute 100 hour capability jobs (jobs exercising at least 90% of the computational capability of the system) to successful completion 95% of the time. If application termination due to system errors can be masked by automatic system initiated parallel checkpoint/restart, then such failures will not count against successful application completion. That is, if the system can automatically take periodic application checkpoints and upon application failure due to system errors automatically restart the application without human intervention, then these interruptions to application progress do not constitute failure of an application to successfully complete.

6.1.5 System Down Time (TR-2)

Over any four week period, the system will have an effectiveness level of at least 95%. The effectiveness level is computed as the average of period effectiveness levels. A new period of effectiveness starts whenever the operational configuration changes (e.g., a component fails or a component is returned to service). Period effectiveness level is computed as University operational use time multiplied by max[0, (p-2d)/p] divided by the period wall clock time. Where p is the number of CPUs in the system and d are the number disabled. Scheduled Preventive Maintenance (PM) is not included in University operational use time.

6.1.6 Scalable RAS Infrastructure (TR-1)

The Offeror will provide a scalable RAS infrastructure that monitors and logs the system health from a centralized control workstation (CWS). All system maintenance functions will be executable from the CWS by the system administration staff.

6.1.6.1 Scalable System Monitoring (TR-1)

All bit errors (from memory, cluster interconnect, local disks, SAN and global disk), over temperature conditions, voltage irregularities, fan speed fluctuations, and disk speed variations shall be logged in the RAS system. All bit errors shall be logged for recoverable and non-recoverable errors. The RAS system will automatically monitor this log constantly, determine irregularities in subsystem function and promptly notify the system administrators. RAS subsystem configuration will include calling out what items are monitored, at what frequency monitoring is done for each item, what constitutes a problem with at least three severity levels (low, medium and high) and notification mechanisms for each item at each severity level.

6.1.6.2 Highly Reliable RAS Infrastructure (TR-1)

The provided scalable RAS infrastructure shall be highly reliable in the sense that there are no single points of failures in the RAS infrastructure and any single component failure does not impact the ability to continue to process the workload on compute, login, I/O and visualization nodes.

6.1.6.3 Failure Isolation Mode (TR-2)

SMP or Node or FRU failures will be determined by supplied diagnostic utilities (not divine intervention), isolated, and routed around without system shutdown. These diagnostic utilities will utilize FRU error detection and fault isolation hardware. Tthe diagnostic utilities will utilize built in error detection and fault isolation circuitry to detect and report failures in all CPU components and in particular the floating-point units. The operators will be able to reconfigure the system to allow for continued operation without use of the failed SMP node or FRU. The capability will be provided to perform this function from a remote network workstation.

6.1.6.4 Scalable System Diagnostics (TR-2)

There will be a scalable diagnostic code suite that checks processor, cache and RAM memory, network functionality, and I/O interfaces for the full system in less than 30 minutes. The supplied diagnostic utilities will quickly, reliably and accurately determine the SMP or Node or FRU failures.

6.1.7 System Graceful Degradation Failure Mode (TR-2)

The failure of a single component such as a single CPU, a single SMP, or a single communications channel will not cause the full system to become unavailable. It is acceptable for the application executing on a failed CPU or SMP to fail but not for applications executing on other parts of the system to fail.

6.1.8 SMP Processor Failure Tolerance (TR-2)

The SMP will be able to run with one or more computational processors disabled, and to do so with minimal performance degradation. That is, the SMP will be able to tolerate failures through graceful degradation of performance.

6.1.9 SMP Memory Failure Tolerance (TR-2)

The Offeror will propose SMPs that are able to run with one or more memory components disabled, and to do so with minimal performance degradation. That is, the SMPs will be able to tolerate failures through graceful degradation of performance where the degradation is proportional to the number of FRUs actually failing.

6.1.10 Replacement Parts and Maintenance (TR-1)

The Offeror will supply hardware and software maintenance for each of the systems for five (5.0) years after acceptance of the systems at LLNL. Software maintenance shall include a trouble reporting mechanism and periodic software updates. In addition, the Offeror shall provide quick turnaround of software fixes to reported bugs. The systems will be installed in a classified area at the Laboratory and so maintenance personnel shall obtain DOE Q clearances.

The University reserves the right to utilize third party memory and hard disk components in any provided nodes. The use of such third party parts by the University shall not prevent Offeror from maintaining the equipment, if maintenance becomes reinstated pursuant to a successful inspection in accordance with Offeror's commercial practice. A maintenance/warranty requalification shall consist of rebooting the nodes and passing the node diagnostics.

6.1.10.1 On-Site Parts Cache (TR-1)

The Offeror will provide a robust on-site parts cache for each system. The robust on-site parts cache will include at least one fully configured compute SMP to be used as a hot-spare and will be sized so that at least five (5) days of spare parts are available, based on Offeror's MTBF statistics for each FRU. The size and content of the complete on-site parts cache and additional complete frame of hot-spare node will be mutually agreed in writing by the University and the Offeror. The University will provide all necessary facilities to accommodate this (these) frame(s) (e.g., floor space, power, cooling, lighting, etc.). The frame(s) will be used for the diagnosis of hardware and/or software problems as necessary. Nodes from this hot-spare frame will be used to replace failing SMPs in the system(s). The hot-swap frame(s) will operate at the SRD classification level and will be administered under DOE security guidelines. Should the agreed to configuration of the hot-swap frame and parts cache prove inadequate, the parties will negotiate an increase in the hot-swap and parts cache configuration. . Additionally, the Offeror will establish a local parts cache which will be derived from approved service planning guidelines but will be not less than thirty (30) days of spare parts based on service planning statistics for each FRU.. A local parts cache is defined as one which requires a round-trip travel time of not more than 1 hour.

6.1.10.2 Response Time and Component Replacement (TR-1)

Hardware maintenance response time shall be less than four (4.0) hours from incident report until Offeror CE personnel begin to repair components identified by this incident report (a repair action). Component failures that take down or render unuseful for ASCI applications more than 50% of the system, will be replaced and brought back to productive status within four hours of CE starting the repair action. Component failures that cause a single node to crash will be repaired in-situ within eight hours of CE starting the repair action. The Offeror personnel shall systematically log incident reports, component failures, CE arrival and repair times and success rate of repair actions within an incident and track multiple incidents from a single failing component. The Offer will verify the validity of these data with University representatives on a weekly basis. If Offeror can not meet these response and repair time requirements for more than

two successive weeks, then Offeror shall refund maintenance costs paid by the University for the period of time the response or repair times are out of specification and take timely corrective action to bring response and repair times within specification.

6.1.11 On-site Analyst Support (TR-1)

The Offeror shall supply three on-site analysts to the University. One on-site systems technician will be highly skilled in operational activities of the proposed EDTV, technology refresh and Purple clusters and shall support University personnel in day-to-day operations and software systems upgrades. The systems technician will use University and Offeror trouble ticket mechanisms and hardware and software problem tracking data bases to maintain an accurate systems availability, calculate actual MTBAF and support response time statistics. One on-site systems programmer will be highly skilled in systems programming and shall support University personnel in providing solutions to the current top ten issues. One on-site applications analyst will be highly skilled in applications development and porting to the EDTV, technology refresh and Purple clusters. This applications analyst will provide expertise to the Tri-Laboratory ASCI code development teams in the areas of software development tools, parallel applications libraries and applications performance. The proposed system will be installed in a classified area at the Laboratory and so analyst personnel shall obtain DOE Q clearances. The University may request additional on-site analysts, which will be priced separately.

6.2 Effective System Performance (ESP) Rating

ESP is designed to evaluate systems for overall effectiveness, independent of processor performance. The ESP test suite simulates "a day in the life of an ASCI platform" by measuring total system utilization under simulated real world conditions. Results take into account both hardware (processor, memory, interconnect, disk) and system software performance. Developed by NERSC as part of a major system procurement process, ESP is designed to predict the effectiveness of a system before purchase, as well as to evaluate system changes before implementation.

The goals of the ESP rating include:

- determine how well an existing system supports a particular scientific workload
- assess systems for that workload before purchase
- provide quantitative effectiveness information regarding system enhancements
- compare different systems on a single workload or discipline
- compare system-level performance on workloads derived from different disciplines
- focus on real world system attributes like job launch and termination, job scheduling for a workload with wildly varying CPU count and runtime requirements and system resiliency under reboot.

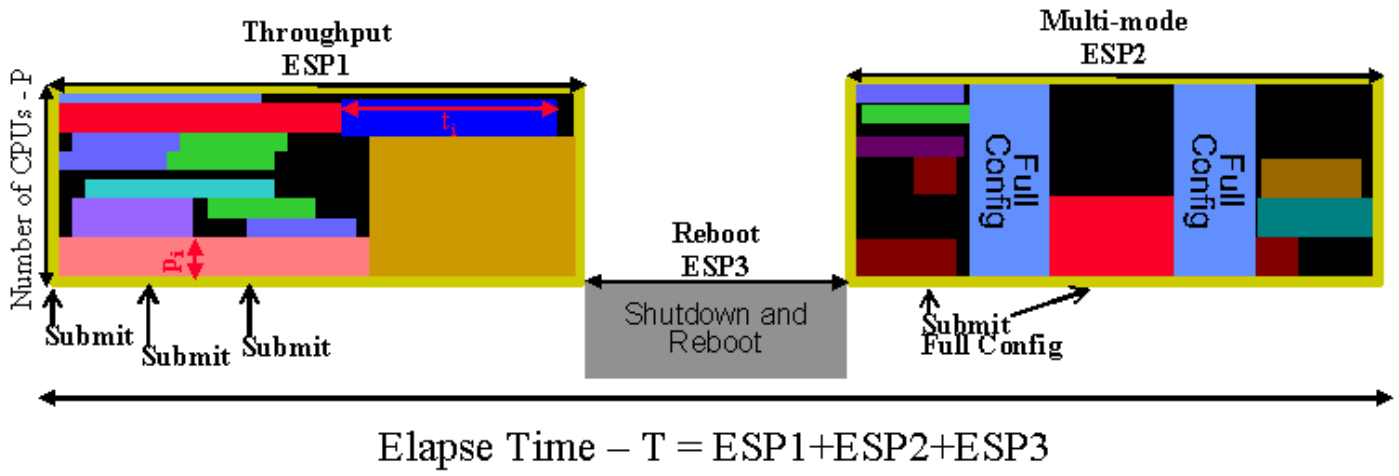


Figure 6.2-1 ESP Timeline

The ESP workload, applications and execution environment are defined in section 9.5. The three ESP components (throughput, multi-mode and reboot) absolute minimum times (AMT1 and AMT2) and observed ESP components (ESP1, ESP2 and ESP3) are defined in section 9.5.3. The Effective System Performance (ESP) is defined there as:

$$ESP = \frac{(AMT1 + AMT2)}{(ESP1 + ESP2 + ESP3)}$$

Since all components of ESP are positive and the numerator is always less then the denominator,

$$0 \le ESP \le 1$$

ESP values closer to unity are better.

Example: Suppose the proposed system has 12,288 CPUs and the CWARM able to schedule the throughput workload at 95% utilization and the multi-mode workload at 90% utilization (quite possible with a fully integrated gang-scheduler utilizing parallel system initiated checkpoint/restart) and the measured ESP reboot time is measured as 2 hours. Then ESP = (3.00+3.06)/(3.00/0.95+3.06/0.90+2.00) = 71%.

6.2.1 Minimum ESP rating (TR-1)

At acceptance, the proposed systems will have a measured effectiveness of $E \ge 70\%$.

6.2.2 Improved ESP rating (TR-1)

The Offeror will improve the effectiveness of the delivered system by 5% annually. The ESP test will be run at least once per year to determine whether or not the required improvement has been attained.

End of Section 6.0

7.0 Facilities

A new facility, known as the Terascale Simulation Facility (TSF), is scheduled to have the first computer floor completion in the time frame of the proposed system delivery (October 2004). **It is assumed that the Offeror will have sufficient facilities to construct the Purple system on-site for shakedown and demonstration, prior to delivery.** This is known as the build-demo-delivery scenario. However, alternate arrangements may be made available if the Offeror does not have these facilities and requires a deliver-build-demo scenario. If Offeror elects the deliver-build-demo scenario, the demo and delivery schedule proposed must reflect TSF building availability dates. The University requires formal written notification of the Offeror's proposed strategy (build-demo-delivery or deliver-build demo or other variant) as soon as possible, but not later than in the RFP response. The Offeror must commit to the proposed strategy with the RFP response. The facilities requirements of Purple are anticipated to be quite significant and the Offeror is admonished to take this aspect of the proposal response extremely seriously.

The TSF first computer room floor will have approximately 125'x190' = 23,750 ft^2 of unobstructed computer floor space, will be able to accommodate system power usage up to 6.0 MW of power at 208V three phase and provide up to 2,000 tons of cooling capacity. This cooling can be used as air cooling, or if required as chilled water with additional facilities modification. All computer power is projected to be fairly reliable, clean, but not conditioned, and there is no UPS. Door opening dimensions and elevator and floor loading capacities will be appropriate for computer installations. A small temporary loading dock will allow for the delivery of equipment, and virtually no space will be available adjacent to the dock for equipment uncrating and staging. This area will lead directly to the computer room space. Uncrating and staging of equipment will have to be done on the computer floor. The computer floor is 48" raised floor with 250 lbs/ft^2 loading. The ceiling above the computer floor has a 10' clearance. The computer floor is on the second floor and use of a 10,000 lbs. freight elevators will be required.

The TSF is being constructed in three stages. The first stage is the foundation and exterior of the building and the full build out of the first computer floor (including required power and cooling), described above. The second stage is the build out of the office tower. The third stage is the build out of the second 23,750 ft^2 computer floor (including additional power and cooling). LLNL will be responsible for supplying the external elements of the power, cooling, and cable management systems.

As much as 8,000-9,000 ft^2 of floor space in the existing B451 computer room "pop-out" are available for the EDTV system. However, some of this space will be required for cooling equipment. There is a maximum of 1.9 MW of power for system usage at 208V three phase and associated cooling available. The computer floor is 30" raised floor with 250 lbs/ft^2 loading. The ceiling above the computer floor has a 10' clearance. Power and cooling requirements for EDTV need to be communicated to the University with the RFP response in as much detail as is available at the time of RFP response in order for the University to plan for EDTV siting and associated facility modifications.

The EDTV and Purple systems will be physically located inside an exclusion area within a security area. The EDTV system will be installed in an existing exclusion area within a security area. Due to the fact that the TSF will be an unclassified work area during construction, the TSF first computer floor will be unclassified work area until after Purple acceptance and stabilization. The TSF first computer floor will become a classified vault type room (VTR) as part of the migration of Purple to classified operations. There will be no dialup capability to classified systems. No remote diagnostics will be allowed. Interaction of the on-site engineering staff with factory support personnel may be limited in some ways (e.g., dissemination of

memory dumps from a classified system may not be allowed). This limitation emphasizes the importance of local access to source code, particularly for operating system daemons. Head disk assemblies (HDAs) from disks used for classified processing cannot be taken off-site or returned to the factory. All on-site personnel will require to be DOE Q-cleared or Q-clearable. It will be extremely difficult to provide access to foreign nationals.

On-site space will be provided for personnel and equipment storage. A safety plan will be required for on-site personnel. They will be expected to practice safe work habits, especially in the areas of electrical, mechanical, and laser activities.

7.1 Power & Cooling Requirements (TR-1)

The Offeror will minimize the power and cooling required by the proposed systems. The Offeror shall provide documentation for the estimated total amount of power in kW (kilowatts) required by the complete EDTV and Purple systems including any subsystems (e.g., disks, external networking, etc.) and the estimated total amount of cooling in BTU (British Thermal Units) required by the complete EDTV and Purple systems. The Offeror shall list separately room air and any liquid cooling required for each system, in relation to the heat load created by operation of each system. The Offeror shall provide monthly updates to such estimates and review the basis for such updated estimates with the University until system installation. In addition, full system estimates for a fully configured system (with maximum amount of SDRAM, I/O subsystems, etc) shall be provided. This documentation shall break down the power and cooling loads to individual frames for each component part of the system.

7.2 Floor Space Requirements (TR-1)

The Offeror will minimize the floor space required by the proposed systems. The Offeror shall provide a floor plan of the proposed EDTV and Purple systems that fits into the space requirements specified below and showing the placement of all system components including, but not limited to: computer frames, input/output device frames, interconnect frames, external networking frames, SAN networking frames, disk device frames.

7.2.1 EDTV Floor Space Requirement (TR-1)

The EDTV system shall be installed in B451 main computer room "pop-out" and shall be less than 8,000 ft^2 as accommodated by this computer floor. This includes, all system components including, but not limited to: computer frames, input/output device frames, interconnect frames, external networking frames, SAN networking frames, disk device frames, and visualization hardware.

7.2.2 Purple Floor Space Requirement (TR-1)

The Purple system shall be installed in B453 first computer room (TSF.1) and shall be less than 125'x190' = 23,750 ft^2. This includes, all system components including, but not limited to: computer frames, input/output device frames, interconnect frames, external networking frames, SAN networking frames, disk device frames, and visualization hardware.

In the event the Offeror has to provide additional hardware to achieve performance requirements specified in the Subcontract, this equipment shall be installed on this TSF.1 floor.

7.3 Installation Plan (TR-2)

The Offeror will provide site installation instructions to the University delineating all site preparation work necessary to install and operate the systems, as configured in this subcontract. These instructions shall delineate the type of electrical equipment required for installation (power couplings and placement, floor loading, etc.). This information will be delivered to the University within 30 days of receipt of contract for EDTV and within six months of receipt of contract for the technology refresh and Purple systems.

End of Section 7.0

8.0 Project Management

The system requirements for this acquisition are unprecedented in terms of capacity and capability. The challenges the Tri-Laboratory community faces in providing a platform on the scale of EDTV and Purple to meet the DOE Stockpile Stewardship programmatic requirements are no less so. Moreover, these challenges are not only technical, but also manifest themselves in the management and administration of the project. All have substantial impact regarding risk and therefore the probability of project success. We recognize that, ultimately, the University is responsible for the successful integration of all the elements, including those acquired from third-parties, academia, and other ASCI-related efforts, to provide the tera-scale computing environment needed to meet the national goals of the Stockpile Stewardship Program. DOE and the University recognize this acquisition as a primary institutional commitment. We expect our partners to help us successfully meet this commitment.

The experience gained by Lawrence Livermore, Los Alamos, and Sandia in the installation of the Red, Blue, White and Q systems has demonstrated that such an activity taxes the resources and management capabilities of even the largest and best-managed organizations.

Some of the lessons we have collectively learned in fielding and integrating those systems into a useful scientific simulation environment include the following:

- The most important lesson we have learned is that this effort, if it is to succeed, must truly be a "partnership" among all involved. While careful mutual planning on the part of the Laboratory and the Industry partner is essential to meeting requirements, unforeseen events and changes are likely. These events can only be successfully dealt with by a partnership that goes beyond an ordinary vendor-customer relationship. It must be one in which teaming, mutual respect, and an honest desire to achieve success is present on the part of everyone involved.
- Changes in a company's technology roadmap can have significant consequences on the success of the project. Whether from development delay or fundamental changes in a company's technology decisions, change is almost inevitable. It is therefore important that such changes be quickly evaluated for their impact on the project. In addition, strategies must be developed and discussed to mitigate technical and scheduling problems.
- Component availability for system manufacture can affect delivery schedules. This is particularly true for new equipment, for which only a limited quantity of components is available. We have also found this to be the case for some older components owing to the large volumes needed for a system of this size, as well as to the commitments lower-tier suppliers may have to other customers.
- Manufacturing, assembly, and QA for a system of this size can tax even the largest companies. It is therefore important to ensure that sufficient capacity, without compromising quality assurance, is available at the times necessary to meet delivery schedules. In addition, the development and systems stress testing of software releases and patches is an on-going problem for ASCI sized systems. This is due to the fact that these systems are usually the largest systems fielded by a vendor by a wide margin. Careful planning of software testing and releases must be done in order to cost effectively test software.
- Significant resources are needed at the factory for pre-delivery staging and testing. We have found it best to perform pre-delivery staging and testing of portions of the system prior to shipment to LLNL to minimize installation problems later. It reduces the number of "DOA" and infant–mortality component failures, helps to ensure correct hardware, software and firmware operation, and allows for execution of company and Laboratory test programs.

We have found that such activity requires the company to provide resources at the factory in the form of floor space, ancillary equipment (i.e., disks, interconnects), and personnel.

- Installation needs to proceed in a logical, coordinated manner. Systems that are shipped without disks, for example, are generally not useable and take up valuable installation resources and floor space. This problem also speaks to the need for outstanding coordination among all elements within a company to ensure that hardware and software availability be coordinated (i.e., when new hardware is available the software to drive it is also available).
- Shipment logistics have an impact. We expect to have only approximately 1500 square feet adjacent to the computer building loading dock to stage deliveries prior to installation in the computer room. This limitation should be taken into account when formulating delivery plans.

It is therefore important to quickly install each shipment as it arrives. Arrangements may be necessary to ensure that sufficient personnel with the appropriate training are available for installation, as well as factory-based resources to assist as needed if problem escalation is warranted. Again, the sheer size and complexity of this system may require that extraordinary measures be taken by the company. Successful installation test completion will be required prior to the initiation of any acceptance test.

- Acceptance testing is an extension of earlier testing. Although the pre-delivery and installation tests will identify many problem prior to acceptance testing, it has been our experience that new problems may surface. The availability of on-site and factory-based resources to correct such problems will be important.
- Stabilizing the system as quickly as possible is programmatically important. It is imperative that the company, the University, and those supplying third-party products work closely to that end. This arrangement will require that applications engineers as well as hardware and system software engineers be on-site to resolve problems. Lab access to the system and third-party source code, although important in earlier and later stages, is critical at this point, and cannot be overemphasized. Again, the unprecedented size and complexity of this system dictate that resources over and above the norm will undoubtedly need to be brought to bear.
- Post-stabilization resource requirements will also be significant. It is easy to underestimate the number of hardware engineers and software and applications analysts with the appropriate experience and training required to maintain high reliability and availability, to make the best use of the system, and to resolve problems quickly.
- It is also easy to underestimate the extent of the necessary spare parts inventory. Because of the classified environment, use of remote diagnostic procedures will not be allowed.

Because of the complexity of this activity, a very strong project plan is of great importance. The Offeror's understanding of our requirements, approach to meeting those requirements, commitment of resources, and attention to cost are critical to the success of the project. In the same vein, the approach to managing this activity is critical. The need to have the support of corporate senior management and a major commitment to a quality assurance plan are also examples of areas critical to the success of the project.

The specific detailed planning and effort tracking and documentation requirements for the development and manufacturing efforts that will be delivered as part of the contract are delineated in sections 8.2, Detailed Purple Plan Of Record

The specific target delivery milestones for the project that will be met are delineated in section 8.3, Project Milestones.

8.1 Performance Reviews (TR-1)

Quarterly performance reviews will be conducted between the Offeror's corporate executives and the University. The Offeror will submit a Quarterly Project Status Report at least five working days before each quarterly review. The report will provide the status of all work breakdown structure tasks and milestones in the critical path. It will also contain narrative descriptions of anticipated and actual problems, solutions, and the impact on the project schedule. Numbered action items will be taken, assigned, logged, and tracked by the Offeror. The minutes of all project reviews will be recorded in detail by the Offeror and provided to the University for approval within 5 working days after the review.

8.2 Detailed Purple Plan Of Record

This project envisions a quantum advance in delivered performance capability for ASCI scientists and engineers. To successfully reach this level of delivered performance the Offeror shall submit, within thirty (30) days of contract award, a full term, highly focused plan of record, per sections 8.2.1, 8.2.2 and 8.2.3, delineating the management, research, development, acquisition, manufacturing, testing, demonstration, delivery, integration and acceptance testing activities to achieve the project goals. The Offeror and the University shall jointly develop a detailed full term project plan of record for University approval. The plan at the time of submission must be accurate and up to date. At a minimum, the full term plan shall contain the following components: management; hardware; software; risk assessment, mitigation and fallback strategies; collaborations. In addition, each year (by the end of the calendar year) the Offeror shall develop a detailed year plan (section 8.2.4) for the next calendar year and track the project during the year with this plan. The full term plan shall be revised on an on-going basis to reflect the changes in the management team, actual development schedule, risk mitigation strategy and shall be submitted for formal review semi-annually at the first and third quarterly meetings. The plan at the time of submission must be accurate and up to date (within ten days of submission). The University shall review the submitted plan and provide the Offeror written comments within two weeks. The Offeror shall revise the plan based on University feedback and resubmit the plan within two weeks of receiving written comments.

8.2.1 Full-Term Project Management Plan (TR-1)

The Offeror and the University will jointly develop a detailed full term project management plan of record for University approval. It is essential that the management plan be kept up to date with respect to changing personnel and company reorganizations and changes in the Offeror's Purple management structure. The plan will contain at least the following components:

- Management teams and structure: The Offeror's Purple project will be managed with two teams: executive team and technical team. The executive team will meet quarterly and have direct input from and feedback to the University. The Offeror's designated NNSA Partnership Executive will meet quarterly with the Lawrence Livermore National Laboratory Director, NNSA Assistant Secretary for Defense Programs, NNSA Deputy Assistant Secretary for Strategic Computing and Simulation to track hardware and system software milestones as well as other strategic partnership issues. The technical team will have quarterly face-to-face meetings, monthly video teleconferences and weekly teleconference calls. The University will develop and revise quarterly a top-ten issues list. The monthly meeting will have a technical focus and go over project status and the action items stemming from the top-ten list. The quarterly meeting will be higher level and go over project status and recent technical issues and accomplishments. The management plan will list the members of the management and technical teams, provide their resumes and list their roles and responsibilities. The management plan will have an

organizational chart of the management and technical teams and lines of reporting to various parts of the company.

- Organization for core team: List the contributing organizations within the company and how they will be coordinated. Provide an organizational chart of the company that depicts these groups and their lines of responsibility. Include hardware R&D, software R&D, productization, field team and applications support, manufacturing, purchasing and quality assurance. Indicate how these areas will be coordinated by the management team.
- Full term project plan and schedule. Provide a Work Breakdown Structure (including milestones) for the project giving at least five levels of detail, as appropriate, with projected start and finish dates and interdependencies of deliverables. This project plan will elaborate on the tasks and milestones committed to in the scaleable systems development section and clearly delineate the project critical path tasks (see below). Provide a Project Schedule that starts at contract award and ends with successful contract termination. The schedule will be developed using the Critical Path Method (CPM) scheduling technique and will utilize the same numbering scheme as the Work Breakdown Structure. The Project Schedule will be placed under configuration control to ensure that all project schedule updates are accomplished in a manner that preserves an audit trail from the original Project Schedule to the current schedule status. The Schedule will contain sufficient detail to ensure that the University and the Offeror can measure progress on an appropriate number of milestones and tasks on any path or on parallel paths to measure progress and to determine the true critical path to project completion.
- Risk reduction plan. In order to meet the project goals and objectives in a timely manner, indicate fall-back strategies that will become operative should delivery schedules not proceed as rapidly as predicted. Indicate additional resources that will be available, if applicable, to the effort in the event that problems develop. Indicate the potential impact to the program and the mitigation plan, should the potential occur. Delineate the problem escalation and resolution path. Risks will be categorized as to their impact (low, medium and high) and to their probability of occurrence (low, medium and high). The risk mitigation strategies will have decision dates specified for executive partnership decisions on the main plan vs. various fall-back strategies.

8.2.2 Full-Term Hardware Development Plan (TR-1)

The hardware (as defined in sections 2.1 and 3.1) full-term development plan shall contain at least the following components:

- CPU Technology. Identify the planned milestones for processor development that lead to those to be deployed in the EDTV and Purple systems. In particular, provide milestones for silicon process development, sampling, engineering quantities, and production quantities for each processor generation between the EDTV and Purple systems.
- SMP Development. Provide the planned tasks and milestones for SMP product development for system generations covered by this contract. Include tasks and milestones for at least the following development areas: memory architecture; cache coherency protocols; ASIC development; performance modeling efforts; applications analysis; functional verification test; system test. Indicate how and when this technology will be inserted at LLNL to meet contract milestones.
- Cluster Interconnect Development. Provide the planned tasks and milestones for cluster interconnect research and development between the EDTV and Purple system generations. Include tasks and milestones for at least the following development areas: switch ASIC development; interface components; cabling components; adapter design; microcode, driver and MPI software development including support for multiple network adapters per node and SMP; functional

verification test; system test. Indicate how and when this technology will be inserted at LLNL to meet contract milestones.

- Parallel I/O Subsystem Development. Provide the planned tasks and milestones for development of parallel I/O subsystem including functional verification and system test. The I/O test plan must delineate component and end-to-end testing. End-to-end testing is defined as starting (or ending) at an LLNL parallel application running on the EDTV and Purple clusters through the parallel I/O libraries down through the transport layers, through the device drivers and RAID hardware to the disks. Include tasks and milestones for at least the following development areas: RAID adapters; SAN networking; disk development; remote I/O devices and links; architecture planning and modeling; development and architecture. Indicate how and when this technology will be inserted at LLNL to meet contract milestones.
- External Network Connections. By external networking, we mean the standards-based networking (e.g., 1000Base-SW and 10 Gigabit Ethernet) to connect the EDTV and Purple clusters to system area networks at LLNL. Include tasks and milestones for at least the following development areas: integration into SAN, TCP/IP off-load engine (TOE) development; adapters; device drivers and protocols supported; EDTV and Purple external networking architecture development and modeling. Indicate how and when this technology will be inserted at LLNL to meet contract milestones.
- System Scalability and Performance Testing. Provide the planned tasks and milestones for the scalability testing of system components. Include development of hardware for reliability, availability and serviceability (RAS).

8.2.3 Full-Term Software Development Plan (TR-1)

The software (as defined in sections 4.and 5) full term project plan shall contain at least the following components:

- Operating System Development. Provide the planned tasks and milestones for operating system development. Include tasks and milestones for at least the following development areas: scaleable SMP support; SMP partitioning; distributed shared memory locality of reference; support for hardware and system performance monitoring; low latency user callable thread mechanism; memory management; full 64-bit support, journaled file systems; reboot time minimization; SMP local gang scheduling; virtual memory support for batch processing (e.g., process swapping and checkpointing); support for I/O partitions; DCE standard tracking; group routing.
- Single System Image Development. Provide the planned tasks and milestones for cluster wide single system image development. Include tasks and milestones for at least the following development areas: POSIX compliant cluster wide file system; cluster single IP network address; file system failover; cluster wide lock management; cluster wide process and POSIX session ID space; virtual memory DMA between SMPs; load balancing and process migration between SMPs; system administration tools for managing the cluster as a single system;
- Parallel I/O Development. Provide the planned tasks and milestones for cluster wide parallel application I/O development. Include tasks and milestones for at least the following development areas: parallel file system and MPI I/O parallel I/O development; performance. Provide tasks and milestones for connecting the CWFS to external systems such as HPSS and Linux production clusters.
- Compiler Development. Provide the planned tasks and milestones for baseline language (FORTRAN 95, C, C++) development. Include tasks and milestones for at least the following development areas: mixed language support; automatic and directed parallelization (OpenMP) of

applications; latency reduction techniques; compiler optimization; migration support (from EDTV to Purple). Indicate any points where compatibility with Fortran 77 applications decreases.

- Message Passing Environment. Provide the planned tasks and milestones for message passing development. Include tasks and milestones for at least the following development areas: bandwidth and latency targets for MPI; MPI standard tracking; integration with debuggers, profilers and performance analysis tools; interoperability to cluster external resources.
- Code Development Tools. Provide the planned tasks and milestones for code development tools development. Include tasks and milestones for at least the following development areas: parallel make, profilers, debuggers, application performance monitoring tools, GUI development for code development tools.
- Cluster Resource Management Support. Provide the planned tasks and milestones for resource management development. Include tasks and milestones for at least the following development areas: hooks for external policy modules; system monitoring tools; cluster wide gang scheduling.
- Fault Tolerance and Containment. Provide the planned tasks and milestones for the development of fault tolerance and gradual degradation of service in the face of component failure features. Include tasks and milestones for at least the following development areas: failure tolerance of CPUs, memory components; SMPs, interconnect, I/O subsystems; error detection vs. retry; scalable system diagnostics.

8.2.4 Detailed Year Plan (TR-1)

Each year (by the end of the calendar year) the Offeror shall develop and submit to the University for review and approval a detailed year plan for the next calendar year. The Offeror shall track the project during the year with this plan. This plan shall be revised on an on-going basis to reflect the changes in the actual development schedule and shall be submitted for formal review quarterly at the quarterly meetings. The plan at the time of submission must be accurate and up to date. At a minimum, the detailed year plan shall contain the following components: Work Breakdown Structure (WBS), Gantt chart, Offeror product Plan of Record line items; I/O test plan, software test plan, system manufacturing and testing plans (in the years with system deliveries) and descriptive narrative. This plan shall cover the hardware (section 8.2.2) and software (section 8.2.3) areas above with more detail and precision.

The University shall review the submitted plan and provide the Offeror written comments within two weeks. The Offeror shall revise the plan based on University feedback and resubmit the plan within two weeks of receiving written comments.

8.3 Project Milestones (TR-1)

Because of the need to meet Stockpile Stewardship Program goals as quickly as possible, the project schedule and milestones are of critical importance. Meeting the following milestones is critical to the success of the project; earlier is much better. In addition, rapid insertion of technology is important. To this end, the University envisions a process where by the systems are delivered, stabilized, accepted, brought under load of science runs (small number of large CPU count and memory footprint, long running jobs with few users active at a time), brought under “limited availability” load (programmatic workload with a limited number of users, typically 5-10 from each of LLNL, LANL and SNL, targeted specifically at achieving programmatic milestones) and finally “general availability” status (general ASCI workload with no limits on the number of or type of work done by user accounts.

The implementation will entail the installation of EDTV, technology refresh and Purple systems at LLNL. Each system will be assembled from individual SMPs that are interconnected with a high-speed, low-latency interconnect supplied by the Offeror. These clusters will be connected to the site's local campus network and to the wide area network that interconnects the three Laboratories. Access to the resources will be provided locally via the site's existing campus networks and remotely through the ASCI-supplied WAN.

The following milestones are provided as a general framework. These milestones include target dates based on ASCI programmatic requirements and anticipated fiscal year funding. These target dates are TR-1 requirements (i.e., not mandatory) and can be modified to more closely match an Offeror's product roadmap. However, there is a significant value to the University and the ASCI program to early delivery of technology and capability. In particular, Purple acceptance in 4QCY04 is highly desirable. The Offeror will provide the University, in the RFP response, a set of milestones for this section and an associated payment schedule that is applicable to the Offeror's proposed development and deployment timeline and methodology. This general framework assumes the build-demo-deliver scenario for fielding EDTV, technology refresh and Purple clusters.

8.3.1 Full-Term Purple Plan of Record (TR-1)

The Offeror will provide a detailed full-term project management plan, and a full-term hardware development and software development plan thirty (30) days after contract award.

8.3.2 EDTV On-Site Support Personnel (TR-1)

Within thirty (30) days after contract award, the Offeror will supply at least three full-time equivalent on-site personnel as set forth in section 6.1.11 with the following job functions:

- One on-site systems technician will provide system administration and day-to-day operations support and be responsible for maintaining an accurate systems availability, MTBF and support response time statistics, as directed by the University;
- One on-site systems programmer will provide solutions to the current top ten issues, as directed by the University;
- One on-site applications analyst will provide expertise to the University code development teams in the areas of software development tools, parallel applications libraries and applications performance.

8.3.3 Early Deployment of Technology Vehicle Demonstration (TR-1)

Prior to shipment, the Offeror will demonstrate an Early Deployment of Technology Vehicle (EDTV) cluster consistent with requirements in sections 3.0, 4.0 and 6.0 that is sized according to the University exercised option. Demonstration and shipment of the EDTV cluster will be accomplished by the end of fourth quarter of calendar year 2002 (4QCY02). Earlier is better. If the University exercises the option at the time of contract award, the Offeror will demonstrate, a month after demonstration of EDTV, the second EDTV system. This milestone is complete when all the EDTV RED and BLACK systems complete the ESP ratings test and the equipment leaves the Offeror's facility.

8.3.4 Early Deployment of Technology Vehicle Acceptance (TR-1)

The Offeror will deliver and install and support an Early Deployment of Technology Vehicle (EDTV) cluster consistent with requirements in sections 3.0, 4.0 and 6.0 that is sized according to the University exercised option. Delivery will be to LLNL in B451 or as directed, by the end of fourth

quarter of calendar year 2002 (4QCY02). Earlier is better. If the University exercises the option at the time of contract award, the Offeror will deliver, a month after delivery of EDTV, the second EDTV system, as directed by the University. This milestone is complete when all the EDTV RED and BLACK systems complete the ESP ratings test. Completion of this milestone starts the five (5.0) year EDTV maintenance clock.

8.3.5 CY03 Plan and Review (TR-1)

The Offeror will provide a detailed plan of activities and deliverables for calendar year 2003 for University review and approval in the fourth quarter of calendar year 2002 (4QCY02).

8.3.6 Technology Refresh (TR-2)

The Offeror will provide hardware and software technology updates between the installation of the EDTV system and the installation of Purple that will significantly improve the hardware and software environment. Hardware technology updates will meet or exceed the following component scaling parameters:

- Memory Size (Byte/FLOP/s) ≥ 0.5
- Globally Addressable User Disk Space (Byte/FLOP/s) ≥ 20
- Memory Bandwidth (Byte/s/FLOP/s) ≥ 1
- Intra-Cluster Network Aggregate Link Bandwidth (Bytes/s/FLOP/s) ≥ 0.1
- Intra-Cluster Network Bi-Section Bandwidth (Bytes/s/FLOP/s) ≥ 0.05
- System Sustained Productive Disk I/O Bandwidth (Byte/s/FLOP/s) ≥ 0.001
- Cluster High Speed External Network Interfaces (bit/s/FLOP/s) ≥ 0.00125

The target delivery date for this technology refresh is first half of calendar 2003 (1HCY03). This milestone is complete when all the technology refresh RED and BLACK systems complete the ESP ratings test.

8.3.7 CY04 Plan and Review (TR-1)

The Offeror will provide a detailed plan of activities and deliverables for calendar year 2004 for University review and approval in the fourth quarter of calendar year 2003 (4QCY03).

8.3.8 Purple Build (TR-1)

The Offeror will build the Purple system at the Offeror facility in accordance with sections 2.0, 4.0 and 6.0. This milestone is complete when all hardware components have been installed, and a majority of the required software is installed and the CWFS is built. The target build date for Purple is second quarter of calendar 2004 (2QCY04).

8.3.9 Purple Demonstration (TR-1)

Prior to shipment, the Offeror will demonstrate the Purple scalable cluster consistent with requirements in sections 2.0, 4.0 and 6.0. Demonstration and shipment of the Purple cluster will be accomplished by the end of third quarter of calendar year 2004 (3QCY04). Earlier is better. This milestone is complete when the Purple system completes the ESP ratings test and the equipment leaves the Offeror's facility.

8.3.10 CY05 Plan and Review (TR-1)

The Offeror will provide a detailed plan of activities and deliverables for calendar year 2005 for University review and approval in the fourth quarter of calendar year 2004 (4QCY04).

8.3.11 Purple Acceptance and Limited Availability (TR-1)

The Offeror will deliver and install and support the Purple scalable cluster consistent with requirements in sections 2.0, 4.0 and 6.0. Delivery will be to LLNL in TSF.1 or as directed, and acceptance accomplished by the end of fourth quarter of calendar year 2004 (4QCY04). Earlier is better. This milestone is complete when the Purple system is fully assembled and functional (including the visualization SMPs and CWFS); satisfies the performance in sections 2.0, 4.0 and 6.0; completes the ESP ratings test. Completion of this milestone starts the five (5.0) year Purple maintenance clock

8.3.12 Purple Limited Availability Status (TR-1)

The Offeror will stabilize the Purple system and bring it into productive usage by the ASCI program as the Tri-Laboratory capability platform for a limited set of ASCI milestone users with capability jobs by the end of the first quarter of calendar year 2005 (1QCY05). This milestone is complete when Purple satisfies the reliability requirements in sections 2.0 and 6.0; completes the ESP ratings test and achieves Limited Availability usage status.

8.3.13 Combined Open EDTV System (TR-1)

After Purple achieves GA status, the Offeror will combine both portions of the EDTV system on the LLNL open network by the end of the second quarter of calendar year 2005 (2QCY05). This milestone is complete when the combined EDTV system is fully assembled and functional (including the visualization SMPs and CWFS); satisfies the performance and reliability in sections 2.0, 4.0 and 6.0; completes the ESP ratings test and achieves General Availability usage status.

8.3.14 CY06 Plan and Review (TR-1)

The Offeror will provide a detailed plan of activities and deliverables for calendar year 2006 for University review and approval in the fourth quarter of calendar year 2005 (4QCY05).

8.3.15 Purple General Availability Status (TR-1)

The Offeror will improve the stability of the Purple system, improve the ESP rating and keep Purple availability and utilization high enough to support a mixed capability and capacity productive usage by the ASCI and SSP program elements by the end of the first quarter of calendar year 2007 (1QCY07). This milestone is complete when Purple satisfies the reliability requirements in sections 2.0 and 6.0; improves ESP rating by at least 10% and achieves General Availability usage status

End of Section 8.0

9.0 Performance of the System

The Purple benchmarks serve three purposes. First, these carefully chosen and developed benchmarks represents a particular subset and/or characteristic of the expected ASCI workload, which consists of solving complex scientific problems using a variety of state-of-the-art computational techniques. Second, the benchmarks provide the Offeror with the opportunity to provide the University concrete data associated with the performance, reliability and scalability of the proposed systems on applications the ASCI program considers programmatically important. In this role, the benchmarks play an essential role in the RFP response evaluation process. Third, the benchmarks will be used as an integral part of the system acceptance tests and ESP rating measurements. As such, these benchmarks represent a critical component of the subcontract.

The benchmark programs described below will be executed by the Offeror for the purpose of measuring the execution performance and compiler capabilities of the proposed systems to the extent defined in this SOW and the individual readme files for each benchmark code. The general requirements and constraints outlined below shall apply to all of the benchmark codes. Additional requirements and/or constraints found in individual benchmark readme files shall apply to that individual benchmark.

Although all of the benchmark results are considered very important and will be carefully analyzed by the University technical proposal evaluation committee, we understand that the Offeror is working with limited resources. The benchmarks are divided into three tiers to give the Offeror a notion of the relative priority of effort to their execution and reporting the University suggests. Tier 1 codes are designated as TR-1 requirements and have special weight with penalties for the failure to report results in the response evaluation as described in the proposal evaluation attachment. Tier 2 codes are designated as TR-2 requirements and have normal weight in the response evaluation. Tier 3 codes are designated as TR-3 requirements and will yield the Offeror additional technical evaluation credit for reporting their results.

ASCI Purple Benchmarks

	4/25/2001	Language				Parallelism		Description
Tier	Code	F77	F95	C	C++	MPI	OpenMP	
1	sPPM	L				P	P	**Marquee performance code**. Science code. Uniform mesh turbulence calculations.
1	UMT2000		L	L		P	P	**Marquee bandwidth code**. Unstructured mesh transport. Updated driver and inner loops. C kernel and Fortran kernel both available, we plan to distribute the C kernel.
1	PRESTA			L		P		MPI stress test.
1	Streams cachebench	L		L				Memory stress test.
1	ParBenCCh				L			C++ kernels framework compiler stress test.
1	ESP Measurement							Effective System Performance measurement. Tests overall cluster software environments effectiveness on complex scientific workload.
2	SMG2000			L		P	P	Scalable MultiGrid. With OpenMP support. Direct Addressing.
2	MDCASK	L				P	P	Molecular dynamics. Programmatic. Scales well
2	IOR-Posix MPIIO			L		P		I/O stress test. Serial and Parallel
3	IRS			L		P	P	Implicit Radiation Solver.
3	SAGE		L			P		AMR Hydro and advection.
3	AZTEC			L		P		A parallel iterative library for solving linear systems. Indirect Addressing.
3	SPHOT	L				P	P	Critically important physics algorithm. Easily scalable, programmatic science with on additional routine. Currently 2D

9.1 Purple Marquee Demonstration Codes

9.1.1 sPPM Marquee Demonstration Code (TR-1)

The sPPM demonstration code is of special interest to ASCI because it solves important hydrodynamics problems using an aggressive and demonstrated highly-efficient SMP cluster implementation of numerical methods relevant to NNSA's Stockpile Stewardship Program. The sPPM demonstration code represents the current state of an ongoing effort which has demonstrated good processor performance, excellent multitasking efficiency, and excellent message passing parallel speedups all at the same time. Its use as a Purple marquee demonstration is to validate the cluster hardware and software correct functioning and stability by demonstrating a high sustained performance across all processors of a cluster of SMPs with a single large application which is of direct interest to the NNSA Stockpile Stewardship Program as well as to scientific simulation in general. It is expected to bring recognition to the ASCI Program, to the Offeror and the University, as well as to the larger scientific high performance computing community.

The University will witness, verify and certify the demonstration. The sPPM code must be compiled into and run as a 64b executable (all virtual memory pointers being 64b). A sustained performance rate will be computed only for the time step update portion of the sPPM code for a problem size of at least 12,000-cubed grid points for a number of time steps sufficient to run for at least one (1.0) hour of wall clock time. In addition, at least one (1) run for at least one (1.0) hour must be made with each sPPM user process utilizing at least 6.0 GiB of memory. The memory requirement for sPPM is approximately 2*NX*NY*NZ + 3(NY+NY+NZ). Where NX, NY and NZ are the total number of zones in the X, Y and Z dimensions, respectively. The exact problem size will necessarily be determined by the actual EDTV, technology refresh and Purple cluster sizes, the available application memory, and the achieved sustained computation rate. The computation rate will be determined based on the actual executed operations as shown below and the elapsed wall clock time required to complete the time steps on the target cluster.

+, -, *	1 FLOP each
/, sqrt	4 FLOPs each
min, max, abs, sign	1 FLOP each

The sPPM marquee demonstration code will be derived from the sPPM code from the Purple benchmark suite. It will be programmed in Fortran95 with some C and will contain no custom assembly language. It will use either single or double precision IEEE arithmetic (or even a mixture). It will use POSIX or OpenMP threads and MPI message passing. It will use general or scientific library routines if they are, or will be, part of a supported library. It will use the initial conditions built into the sPPM code without the image or texture maps. The time for initialization, visualization and restart dumps will not be included in the demonstration rate determination - visualization and restart dumps could even be disabled. The timing prints per double time step can also be disabled but the one line "courant and energy" print is required on at least node 0 to either STDOUT or to the output file.

Optimizations will be allowed so long as they don't specialize the functionality represented by the reference sPPM benchmark implementation. In particular, the source code for the hydro kernel of the sPPM benchmark code (i.e. the file sppm.m4) will be used as provided by the University (i.e. unmodified), except for the addition of compiler directives. The University will continue its efforts

to improve the efficiency of the code. Even for the subroutines in sppm.m4, tuning can be achieved through selecting alternate pieces of provided code through preprocessor flags, through the parameter IQ, etc. The goal is to emphasize higher level optimizations as well as compiler optimization technology improvements while maintaining *readable* code and physics modules. Higher level optimizations and compiler optimizations will be allowed so long as they don't increase the runtime or artificially increase the delivered FLOP/s rate by performing non-useful work. One obvious permitted modification in the parallel part of the sPPM demonstration code will be the overlapping of computation and communication (i.e., asynchronous communication).

Ideas to demonstrate Tera-scale performance levels on additional ASCI relevant applications will be suggested, but only if such opportunities arise and prove feasible and do not interfere with the sPPM and UMT2000 Purple demonstrations.

The Subcontractor will provide a reasonable effort to obtain a sustained performance of 30% of peak floating-point execution rate across the entire EDTV, technology refresh and Purple systems on the sPPM benchmark. Reasonable effort is at least two person years of effort staffing and managing a working group focused on sPPM optimization and having those optimizations then integrated into the baseline languages (participants include representatives from the University and the Offeror's research, compiler development, communications subsystem and application support groups).

9.1.2 UMT2000 Marquee Demonstration Code (TR-1)

The UMT benchmark is a 3D, deterministic, multigroup, photon transport code for unstructured meshes. The transport code solves the first-order form of the time-dependent Boltzmann transport equation. This code was chosen because the core computational techniques are congruent with those that are anticipated to burn most of the computer cycles achieving the ASCI applications milestones over the lifetime of the machine. Due to the complex data structures utilized, UMT2000 performance is dominated by delivered memory bandwidth on a single node and MPI bandwidth for long and short messages in the cluster. This application also demonstrates the standard ASCI programming methodology: an application implemented in both Fortran95 and C with OpenMP thread parallelism on a node and MPI between nodes in the cluster. Its use as a Purple marquee demonstration is to validate the cluster hardware and software correct functioning and stability for a code that stresses the memory and communications subsystems with mixed language and parallelism implementation. The target performance demonstration is a moderate percentage of peak across all processors of a cluster of SMPs with a single large application which is of direct computational relevance to ASCI applications milestones as well as general of great importance to NSSA's Stockpile Stewardship Program. It is expected to bring recognition to the ASCI Program, to the Offeror and the University, as well as to the larger scientific high performance computing community.

The MPI-based parallelism in the Fortran portion uses mesh decomposition to distribute the mesh across the specified MPI tasks. The OpenMP based parallelism in the C kernel then divides the ordinates among the OMP threads. This C kernel's computation time typically completely dominates the execution time of the benchmark.

The University will witness, verify and certify the demonstration. The UMT2000 code must be compiled into and run as a 64b executable (all virtual memory pointers being 64b). A sustained performance rate will be computed only for the time step update portion of the UMT2000 code for a problem sized to utilize at least 80% of available memory for a number of time steps sufficient to run for at least one (1.0) hour of wall clock time. The exact problem size will necessarily be determined

by the actual EDTV, technology refresh and Purple cluster sizes, the available application memory, and the achieved sustained computation rate. In addition, at least one (1) run for at least one (1.0) hour must be made with each UMT2000 user process utilizing at least 6.0 GiB of memory. The computation rate will be determined based on the actual executed operations as shown below and the elapsed wall clock time required to complete the time steps on the target cluster.

+, -, *	1 FLOP each
/, sqrt	4 FLOPs each
min, max, abs, sign	1 FLOP each

The UMT2000 marquee demonstration code will be derived from the UMT2000 code from the Purple benchmark suite. It will be programmed in Fortran95 and C and will contain no custom assembly language. It will use double precision IEEE arithmetic. It will use OpenMP threads and MPI message passing. It will use general or scientific library routines if they are, or will be, part of a supported library. It will use the initial conditions UMT2000 in the input files supplied with the code. The time for initialization will not be included in the demonstration rate determination - only the performance in the core transport calculation will be considered.

Optimizations will be allowed so long as they don't specialize the functionality represented by the reference UMT2000 benchmark implementation. In particular, the source code for the transport kernel of the UMT2000 benchmark code (i.e. the file snswp3d.c) will be used as provided by the University (i.e. unmodified), except for the addition of compiler directives. The University will continue its efforts to improve the efficiency of the code. The goal is to emphasize higher level optimizations as well as compiler optimization technology improvements while maintaining *readable* code and physics modules. Higher level optimizations and compiler optimizations will be allowed so long as they don't increase the runtime or artificially increase the delivered FLOP/s rate by performing non-useful work. One obvious permitted modification in the parallel part of the UMT2000 demonstration code will be the overlapping of computation and communication (i.e., asynchronous communication in rtmain.f and exchange.f).

The Subcontractor will provide a reasonable effort to obtain a sustained performance of 15% of peak floating-point execution rate across the entire EDTV, technology refresh and Purple systems on the UMT2000 benchmark. Reasonable effort is at least two person years of effort staffing and managing a working group focused on UMT2000 optimization and having those optimizations then integrated into the baseline languages (participants include representatives from the University and the Offeror's research, compiler development, communications subsystem and application support groups).

9.2 Benchmark Suite

The benchmark programs described below will be executed by the Offeror for the purpose of measuring the execution performance and compiler capabilities of the reference system to the extent defined in the benchmark readme file for each code. Each of the benchmark programs represents a particular subset and/or characteristic of the expected ASCI workload, which consists of solving complex scientific problems using a variety of state-of-the-art computational techniques. The general requirements and constraints outlined below shall apply to all of the benchmark codes. Additional requirements and/or constraints found in individual benchmark readme files shall apply to that individual benchmark.

The benchmark programs are available via the Web at the following URL:
http://www.llnl.gov/asci/purple/benchmarks/
The individual benchmark codes can be downloaded as tar files. There are two readme files for most benchmarks. The first provides general information about that benchmark including a description of the code, how to build and run it, and any specific information about timing or storage issues. The second readme file contains benchmark specific instructions and constraints. It will also contain expected runs and results, modification record, and RFP formal questions and answers. Two of the benchmark codes require licensing paperwork be completed to gain access, no cost is involved.

The benchmark suite will consist of two "marquee codes" or challenge apps, sPPM and UMT1.2 (UMT2000). These application codes will be released with the draft RFP and will have specific performance levels associated that must be met. In addition to the challenge apps, there will be an additional seven application codes that must be run on the reference system and will be involved in later acceptance of the machine. There will also be three stress test runs required: MPI, Memory, and I/O.

The sPPM benchmark solves a 3D gas dynamics problem on a uniform Cartesian mesh, using a simplified version of the PPM (Piecewise Parabolic Method) code -- hence the "s" for simplified. The code is written to simultaneously exploit OpenMP threads for multiprocessing shared memory parallelism and domain decomposition with message passing for distributed parallelism. It represents the current state of ongoing research that has demonstrated good processor performance, excellent multi-threaded efficiency, and excellent message passing parallel speedups all at the same time. Performance optimizations to the sPPM source are encouraged so long as they don't specialize the functionality. However, the hydro kernel (i.e., the sppm.m4 source file) must be used UNMODIFIED except for the insertion of compiler directives. Alternative coding can be selected through the m4 defines (ifsupr, ifcvmg, ifinln, and ifcray) found in the sppm.h file. Also note that changes outside of sppm.m4 can affect it. For example, it is possible to use the IQ parameter in iq.h to affect the array sizes in these routines. The goal is to emphasize higher-level optimizations and/or compiler technology instead of relying on hand tweaked specialized code. The benchmark must use MPI message passing between SMPs. It cannot overlap the boundary communications in the BDRYS routines (in bdrys.m4) with the directional sweeps in the CALCHYD routines (in runhyd3.m4) since we want to measure communication separately; however, overlapped communication and computation will be allowed for the EDTV demonstration. The benchmark is expected to use the OpenMP multi-threaded shared memory parallelism strategy it now uses. The sPPM benchmark must use double precision (64b) floating-point arithmetic and be a 64b executable (utilize 64b virtual memory addressing); the EDTV, technology refresh and Purple demonstration will allow either single (32b) or double precision (64b) floating-point arithmetic, but all demonstration runs must be 64b executables.

The UMT2000 benchmark is a 3D deterministic, multi-group, photon transport code for unstructured meshes. The transport code solves the first-order form of the time-dependent Boltzmann transport equation. The energy dependence is modeled using multiple photon energy groups. The angular dependence is modeled using a collocation of discrete directions, or "ordinates." The spatial variable is modeled with an "upstream corner balance" finite volume differencing technique. The solution proceeds by tracking through the mesh in the direction of each ordinate for each energy group, accumulating the desired solution on each zone in the mesh. Hence, memory access patterns may vary substantially for each ordinate on a given mesh and the entire mesh is "swept" multiple times. The purpose of this application is to provide an example of unstructured mesh transport. In the process of optimization the mesh construction code should not be changed. Because the runtime of the application is overwhelmingly dominated by the kernel routines that implement the ordinate direction and energy

group sweeps, UMT scales well to very large processor counts. This computationally "expensive" kernel is inside the OpenMP loop, providing near-optimal conditions for high OpenMP efficiency. This code has been successfully run on up to 1,600 processors.

The UMT2000 benchmark must use double precision (64b) floating-point arithmetic and be a 64b executable (utilize 64b virtual memory addressing). The EDTV, technology refresh and Purple demonstration must be double precision (64b) floating-point arithmetic and all demonstration runs must be 64b executables.

The remaining code descriptions can be found in their respective general README files.

9.3 System Configuration (TR-1)

The reference benchmark system shall be a scaled down version of the proposed EDTV system. The reference benchmark system shall be at least a one teraFLOP/s peak configuration consisting of at least two SMPs and a total of at least 288 processors. The reference CWFS should be at least 2.0 TB and architected to deliver a minimum of 0.5 GB/s. The reference benchmark system used should be fully described as part of the benchmark response. The reference system will also require a parallel file system to respond to the I/O portion of the test. Alternate benchmarking configurations may be utilized after discussion with the University on the benchmarking strategy and relevance of the results to the proposed systems.

The benchmark system should contain the same processors, cache, memory, SMP size, interconnects, I/O subsystem, etc., that is proposed for the EDTV system. If this is not possible, benchmark results from an alternative system that meets the conditions specified in the previous paragraph may be reported but the Offeror shall also provide estimated scaled performance for the EDTV configuration consistent with the benchmark system configurations as identified in the previous paragraph. All scaling arguments should be fully described by the Offeror in the response and will be reviewed and evaluated by the University; supporting documentation may be provided. The University will be the sole judge of the validity of any scaled results.

9.4 Test Procedures (TR-1)

The benchmark runs will be made according to the following test procedures. The ASCI systems will be primarily used in a high-level language environment. It is the intent of these benchmarks to measure performance of the reference system from this standpoint. Recoding of the benchmarks or portions of the benchmarks in assembly language is prohibited. The use of library routines that currently exist in an Offeror's supported set of general or scientific libraries, or will be in such a set when the EDTV and full-scale systems are delivered, is allowed at the University's discretion when they do not specialize or limit the applicability of the benchmark nor violate the measurement goals of the particular benchmark. Source preprocessors, execution profile feedback optimizers, etc. are allowed as long as they are, or will be, available and supported as part of the compilation system for the EDTV and full-scale systems. All benchmarks will be run in double precision (64b) floating point arithmetic and as 64b executables (64b virtual memory addressing). All benchmarks that use the message-passing programming paradigm will use a supported 64b virtual memory pointer, thread safe communication library that implements the MPI standard. All benchmarks that use the threads programming paradigm will use a supported communication library that implements the OpenMP standard. MPI and OpenMP functionality must be simultaneously usable by single application codes. Some specific constraints about the use of library

routines apply to individual benchmarks as stated in individual benchmark readme files. The required run configurations for each benchmark will be described in the individual benchmark readme files. OpenMP based parallelism should be utilized to the extent possible on each SMP platform. Each SMP platform will be a set of CPU's sharing random access memory within the same memory address space.

Changes to accommodate unique hardware and software characteristics of a system that are consistent with the preceding paragraph will be allowed except where specifically prohibited in the constraints for each benchmark. Code modifications will be documented in the form of initial and final source files, with mandatory accompanying text describing the changes. An audit trail will be supplied to the University for any changes made to the benchmark codes. The audit trail will be sufficient for the University to determine that changes made violate neither the spirit of the benchmark nor the specific restrictions on the various benchmark codes. We require that all benchmark codes first be run as provided, without any code modifications, in each required configuration and that these baseline results be included along with any results obtained from modified code.

Output and measurements that will be provided by the Offeror for each benchmark are:
1. The output of the results generated by each individual benchmark run,
2. Any additional performance measurements as requested in the readme of each benchmark,
3. The CPU time, system time, and wall clock time for the entire execution of each individual benchmark run,
4. All compilation options, the wall clock time required to compile all source code, and the wall clock time to load/link (i.e., create an executable image) for each benchmark. Each compiler option used must have a short description of the purpose of the option.
5. All environment variables and any other system or user settings during execution.

Correct execution and measurements will be certifiable by the University.

In addition to the results obtained for each benchmark on the reference system, we expect the Offeror to provide estimated scaled performance figures for each benchmark to the EDTV system and to the final full-scale system. All scaling arguments will be fully described by the Offeror and will be reviewed and evaluated by the University; supporting documentation may be provided. The University will be the sole judge of the validity of any scaled results.

The selection will be based on the complete RFP response including cost, when, what, and who. All benchmark results and information will be judged as an integral part of the "what" selection criteria. This is described further in the Evaluation Criteria section of the SOW (Attachment 4).

9.5 ESP Measurements (TR-1)

The Effective System Performance (ESP) test provides a performance metric for production oriented parallel systems. It measures the utilization of the system in a typical operating environment that includes multiple jobs of different size and duration, asynchronous job submission, changing of operation modes (e.g. from debugging and testing to full capability) and system management tasks. It focuses on operating system attributes that are used to efficiently execute applications such as parallel launch time, job scheduling, preemptive job launch and other system management functions. The primary objective of the ESP test is to run a fixed number of parallel jobs through a batch scheduler in the minimum elapsed time.

The metric is independent of processor-speed, system size, application, and compiler functionality and has only modest contention for shared system resources (e.g. file systems). As such it is different from a

throughput test although the mechanics of the test are similar. Further discussion of the ESP test is available in the SC2000 Proceedings and at <http://www.nersc.gov/aboutnersc/esp.html>

The individual jobs in the test are specifically tailored such that the elapsed run times closely approximate target run times. There are 230 jobs comprised from a list of 14 job types. The "14 types" are derived from only 3 benchmarks and represent jobs of different sizes and run lengths. Table 9-1 below lists the job types with their fractional-size, instance count and target run times. See the article referenced above for further details on the origin and design of this table. The ESP measurement is divided into three components: throughput, multi-mode and reboot. The muti-mode variant runs all the jobs listed below on the fractional-size of the system with the target run-times with a specific number of resubmissions (count). The throughput workload is identical, except that the Z jobs are not run. The reboot variant measures the time it takes to shutdown, reboot the system and bring it back to full usefulness.

				System with 288 CPU			System with 12,288 CPU		
Job-type	Fract-Size	Count	Target Run Time	Job CPU Count	Job Work	Total Work	Job CPU Count	Job Work	Total Work
A	0.03125	75	267	9	2,404	180,225	384	102,528	7,689,600
B	0.06250	9	322	18	5,796	52,164	768	247,296	2,225,664
C	0.50000	3	534	144	76,896	230,688	6,144	3,280,896	9,842,6883
D	0.25000	3	616	72	44,352	133,056	3,072	1,892,352	5,677,056
E	0.50000	3	315	144	45,360	136,080	6,144	1,935,360	5,806,080
F	0.06250	9	1846	18	33,228	299,052	768	1,417,728	12,759,522
G	0.12500	6	1334	36	48,024	288,144	1,536	2,049,024	12,294,144
H	0.15820	6	1067	45	48,015	288,090	1,944	2,074,248	12,445,488
I	0.03125	24	1432	9	12,888	309,312	384	549,888	13,197,312
J	0.06250	24	725	18	13,050	313,200	768	556,800	13,336,200
K	0.09570	15	487	27	13,149	197,235	1,176	572,712	8,590,680
L	0.12500	36	366	36	13,176	474,336	1,536	562,176	20,238,336
M	0.25000	15	187	72	13,464	201,960	3,072	574,464	8,616,960
Z	1.00000	2	~100	288	28,800	57,600	12,288	1,228,800	2,457,600
	Total	230		936	398,601	2,161,142	33,984	17,044,727	135,204,360
AMT2 (sec)					1,384.03	10,976.19		1,387.07	11,002.96
AMT2 (HR)						3.05			3.06
AMT1 (sec)					1,284.03	10,776.19		1,287.07	10,802.96
AMT1 (HR)						2.99			3.00

Table 9-1: Jobs for ESP

The fractional-size is simply the size of the job as a fraction of total system size. For example, if the benchmarking system has 288 processors for computation, then the size of job-type B is 18 (0.06250 x 288) processors.

The fractional-size is simply the size of the job as a fraction of total system size. For example, if the benchmarking system under test has 288 processors for computation, then the size of job-type B is 18 (0.06250 x 288) processors. Thus, the ESP test (version 2) can be applied to any system size and has been verified on 64, 512 and 2,048 processor systems. For the purposes of this discussion, it is useful to define a unit of computational "work" as the product of the run time and job size (number of processors). Following our example, job-type B is designated 18 CPU x 322 secs = 5,796 CPU seconds of work. Therefore, for a 288 processor benchmarking system, the total amount of work in the ESP test is 288 * sum (Fract-size * run time * count) = 2,161,142 CPU seconds. Given a total amount of work, a hypothetical absolute minimum time, (AMT), can be computed by dividing the work by the system size. In this case, AMT2 = 10,976 seconds (approximately 3 hours) for the multi-mode (with Z jobs) and AMT1 = 10,776 for the throughput (without Z jobs). Note that the AMT is independent of the total system size. A second example is given in Table X for a system with 12,288 CPUs. The Effective System Performance (ESP) is defined as:

$$ESP = \left(AMT1 + AMT2\right) \Big/ \left(ESP1 + ESP2 + ESP3\right)$$

Since all components of ESP are positive and the numerator is always less then the denominator,

$$0 \le ESP \le 1$$

This is the key result of the ESP test. For increasingly efficient systems, the ratio approaches unity.

9.5.1 ESP Modifications

No modifications to the three (3) applications are allowed except where necessary for code porting and correct results on the Offeror's system. Offerors are expected to change the input data for each application as necessary to reach the required run times of the test. Submission scripts can be modified to the extent necessary to make the scripts compatible to the Offeror's batch control system. None of these modifications should compromise the intent of the test. Further clarification may be found in the test README file.

9.5.2 Execution Requirements

The three ESP measurements will be made on the benchmarking system described in section 9.3. The first component of the ESP test requires executing the job mix described by Table 9-1 except job-type Z (228 jobs). This is designated the “throughput” variant and is listed in the tables below as ESP1. All the jobs are submitted at the beginning of the test in an order determined by a fixed pseudo-random sequence.

The second component of the test, called the “multi-mode” variant and listed in the tables below as ESP2, is identical to the "throughput" variant except the two Z jobs are submitted at 2,400 and 7,200 seconds after the start of the test and after all other jobs, A to M, have been submitted. The Z jobs must be launched as soon as possible after submission. That is, no other queued job is permitted to start while there is a Z job in the queue. As a first step, this may be accomplished by assigning the Z jobs high priorities. How to further expedite the Z job depends on how the system handles running jobs. Some of the options include roll-out, checkpoint or suspension. As a last resort, one can simply drain the system of running jobs until the Z job fits. The ESP test does not mandate how this is achieved, simply that no other job is permitted to start running in the interim between the submission

of the Z job and its launch. Manual intervention (or hand scheduling) is not permitted once the test has been initiated.

The third component of the test, called "reboot" variant and listed in the tables below as ESP3, represents the most common wasted time required in most basic system administration activities: the time it takes to shutdown and boot the entire system. The "reboot" begins by issuing the command to stop all work on a fully loaded system and idle the system, followed by the commands to reboot the entire system and restore it totally to a running state, including bringing up the external network and CWFS, CWARM and other system features and is begins to schedule the first job in a previously queued workload. Note this does not require equipment to be power cycled. Also note that this does not have to be a full cold boot in the sense that all hardware need not be automatically sensed for changes and checked upon reboot. Previously sensed and tested hardware runtime environment can be reused. However, all nodes must return to full operating condition with at least all memory, local disk and global disk, interconnect available for application use and system services functioning.

9.5.3 Reporting

The Offeror shall report in Table 9-2, Table 9-3 and Table 9-4 the elapsed wall clock of the test (in seconds) and project the times for all system configurations and computed AMTs proposed (in seconds) for the three components of the ESP test. The Offeror shall fully document the methodology by which these projections are made.

Component	Absolute Minimum Time	Elapsed Wall Clock Time (secs)
Throughput variant	AMT1 = 10,776	ESP1 =
Multi-mode variant	AMT2 = 10,976	ESP2 =
Reboot	N/A	ESP3 =

Table 9-2: ESP – Observed Timings on Benchmark System

Component	Absolute Minimum Time	Elapsed Wall Clock Time (secs)
Throughput variant	AMT1 = 10,776	ESP1 =
Multi-mode variant	AMT2 = 10,976	ESP2 =
Reboot	N/A	ESP3 =

Table 9-3: ESP – Projected Timings for EDTV.

Component	Absolute Minimum Time	Elapsed Wall Clock Time (secs)
Throughput variant	AMT1 = 10,776	ESP1 =
Multi-mode variant	AMT2 = 10,976	ESP2 =
Reboot	N/A	ESP3 =

Table 9-4: ESP – Projected Timings for Purple.

End of Section 9.0

10.0 Appendix A Glossary

10.1 Hardware

b	bit. A single, indivisible binary unit of electronic information.
B	Byte. A collection of eight (8) bits.
32b floating-point arithmetic	Executable binaries (user applications) with 32b (4B) floating-point number representation and arithmetic. Note that this is independent of the number of bytes (4 our 8) utilized for memory reference addressing.
32b virtual memory addressing	All virtual memory addresses in a user application are 32b (4B) integers. Note that this is independent of the type of floating-point number representation and arithmetic.
64b floating-point arithmetic	Executable binaries (user applications) with 64b (8B) floating-point number representation and arithmetic. Note that this is independent of the number of bytes (4 our 8) utilized for memory reference addressing.
64b virtual memory addressing	All virtual memory addresses in a user application are 64b (8B) integers. Note that this is independent of the type of floating-point number representation and arithmetic. Note that all user applications should be compiled, loaded with Offeror supplied libraries and executed with 64b virtual memory addressing by default.
CE	On-site hardware customer engineer performing hardware maintenance with DOE Q-clearance.
Cluster	A set of SMPs connected via a scalable network technology. The network will support high bandwidth, low latency message passing. It will also support remote memory referencing.
CPU	Central Processing Unit or processor. A VLSI chip constituting the computational core (integer, floating point, and branch units), registers and memory interface (virtual memory translation, TLB and bus controller).
CWFS	Cluster Wide File System. The file system that is visible from every node in the system with scalable performance. See section 2.2.1.13.
FLOP or OP	Floating Point OPeration.
FLOPS or OPS	Plural of FLOP.
FLOP/s or OP/s	Floating Point OPeration per second.
GB	gigaByte. gigaByte is a billion base 10 bytes. This is typically used in every context except for Random Access Memory size and is 10^9 (or 1,000,000,000) bytes.
GiB	gibiByte. gibiByte is a billion base 2 bytes. This is typically used in terms of Random Access Memory and is 2^{30} (or 1,073,741,824) bytes. For a complete description of SI units for prefixing binary multiples see URL: http://physics.nist.gov/cuu/Units/binary.html
GFLOP/s or GOP/s	gigaFLOP/s. Billion (10^9 = 1,000,000,000) 64-bit floating point operations per second.

MB	megaByte. megaByte is a million base 10 bytes. This is typically used in every context except for Random Access Memory size and is 10^6 (or 1,000,000) bytes.
MiB	mebiByte. mebiByte is a million base 2 bytes. This is typically used in terms of Random Access Memory and is 2^{20} (or 1,048,576) bytes. For a complete description of SI units for prefixing binary multiples see URL: http://physics.nist.gov/cuu/Units/binary.html
MFLOP/s or MOP/s	megaFLOP/s. Million (10^6 = 1,000,000) 64-bit floating point operations per second.
Mpixel	megapixel. Million (10^6 = 1,000,000) pixels.
Mpolygons	megapolygon. Million (10^6 = 1,000,000) polygon.
MTBF	Mean Time Between Failure. A measurement of the expected reliability of the system or component. The MTBF figure can be developed as the result of intensive testing, based on actual product experience, or predicted by analyzing known factors. See URL: http://www.t-cubed.com/faq_mtbf.htm
Node	Independent operating system partition on an SMP.
NUMA	Non-Uniform Memory Access architecture. The distance in processor clocks between processor registers depends on where in main memory the address points to. That is, a load/store operation latency for some memory locations is larger than that for others.
Peak Rate	The maximum number of 64-bit floating point instructions (add, subtract, multiply or divide) per second that could conceivably be retired by the system. For RISC CPUs the peak rate is typically calculated as the maximum number of floating point instructions retired per clock times the clock rate.
Pixel	The smallest image-forming unit of a video display.
Polygon	A closed plane figure bounded by three or more line segments. Aggregations of polygons in three dimensional space are commonly used in computer visualization as a simplification to represent more complicated (smooth) three dimensional shapes.
Scalable	A system attribute that increases in performance or size as some function of the peak rating of the system. The scaling regime of interest is at least within the range of 1 teraFLOP/s to 60.0 (and possibly to 120.0) teraFLOP/s peak rate.
SMP	Shared memory Multi-Processor. A set of CPUs sharing random access memory within the same memory address space. The CPUs are connected via a high speed, low latency mechanism to the set of hierarchical memory components. The memory hierarchy consists of at least processor registers, cache and memory. The cache will also be hierarchical. If there are multiple caches, they will be kept coherent automatically by the hardware. The main memory may be a Non-Uniform Memory Access (NUMA) architecture. The access mechanism to every memory element will be the same from every processor. More specifically, all memory operations are done with load/store instructions issued by the CPU to move data to/from registers from/to the memory. A single SMP may be partitioned into one or more nodes.

Tera-Scale	The environment required to fully support production-level, realized teraFLOP/s performance. This environment includes a robust and balanced processor, memory, mass storage, I/O, and communications subsystems; robust code development environment, tools and operating systems; and an integrated cluster wide systems management and full system reliability and availability.
TB	TeraByte. TeraByte is a trillion base 10 bytes. This is typically used in every context except for Random Access Memory size and is 10^{12} (or 1,000,000,000,000) bytes.
TiB	TebiByte. TebiByte is a trillion bytes base 2 bytes. This is typically used in terms of Random Access Memory and is 2^{40} (or 1,099,511,627,776) bytes. For a complete description of SI units for prefixing binary multiples see URL: http://physics.nist.gov/cuu/Units/binary.html
TFLOP/s	teraFLOP/s. Trillion (10^{12} = 1,000,000,000,000) 64-bit floating point operations per second.
UMA	Uniform Memory Access architecture. The distance in processor clocks between processor registers and every element of main memory is the same. That is, a load/store operation has the same latency, no matter where the target location is in main memory.

10.2 Software

32b executable	Executable binaries (user applications) with 32b (4B) virtual memory addressing. Note that this is independent of the number of bytes (4 our 8) utilized for floating-point number representation and arithmetic.
64b executable	Executable binaries (user applications) with 64b (8B) virtual memory addressing. Note that this is independent of the number of bytes (4 our 8) utilized for floating-point number representation and arithmetic. Note that all user applications should be compiled, loaded with Offeror supplied libraries and executed with 64b virtual memory addressing by default.
API (Application Programming Interface)	Syntax and semantics for invoking services from within an executing application. All APIs will be available to both Fortran and C programs, although implementation issues (such as whether the Fortran routines are simply wrappers for calling C routines) are up to the supplier.
Current standard	Term applied when an API is not "frozen" on a particular version of a standard, but will be upgraded automatically by Offeror as new specifications are released (e.g., "MPI version 2.0" refers to the standard in effect at the time of writing this document, while "current version of MPI" refers to further versions that take effect during the lifetime of this contract.
Fully supported (as applied to system software and tools)	A product-quality implementation, documented and maintained by the HPC machine supplier or an affiliated software supplier.

Gang Scheduling	When a user job is scheduled to run, the gang scheduler must contemporaneously allocate to CPUs all the threads and processes within that job (either within an SMP or within the cluster of SMPs). This scheduling capability must control all threads and processes within the SMP cluster environment.
Job	A job is a cluster wide abstraction similar to a POSIX session, with certain characteristics and attributes. Commands will be available to manipulate a job as a single entity (including kill, modify, query characteristics, and query state). The characteristics and attributes required for each session type are as follows: 1) interactive session: an interactive session would include all cluster wide processes executed as a child (whether direct or indirect through other processes) of a login shell and will include the login shell process as well. Normally, the login shell process will exist in a process chain as follows: init, inetd, [sshd | telnetd | rlogind | xterm | cron], then shell. 2) batch session: a batch session will include all cluster wide processes executed as a child (whether direct or indirect through other processes) of a shell process executed as a child process of a batch system shepherd process, and will include the batch system shepherd process as well. 3) ftp session: an ftp session would include an ftpd and all its child processes. 4) kernel session: all processes with a pid of 0. 5) idle session: this session does not necessarily actually consist of identifiable processes. It is a pseudo-session used to report the lack of use of resources. 6) system session: all processes owned by root that are not a part of any other session.
Published (as applied to APIs):	Where an API is not required to be consistent across platforms, the capability lists it as "published," referring to the fact that it will be documented and supported, although it will be Offeror- or even platform-specific.
Single-point control (as applied to tool interfaces)	Refers to the ability to control or acquire information on all processes/PEs using a single command or operation.
Standard (as applied to APIs)	Where an API is required to be consistent across platforms, the reference standard is named as part of the capability. The implementation will include all routines defined by that standard (even if some simply result in no-ops on a given platform).
XXX-compatible (as applied to system software and tool definitions)	Requires that a capability be compatible, at the interface level, with the referenced standard, although the lower-level implementation details will differ substantially (e.g., "NFSv4-compatible" means that the distributed file system will be capable of handling standard NFSv4 requests, but need not conform to NFSv4 implementation specifics).

End of Section 10.0